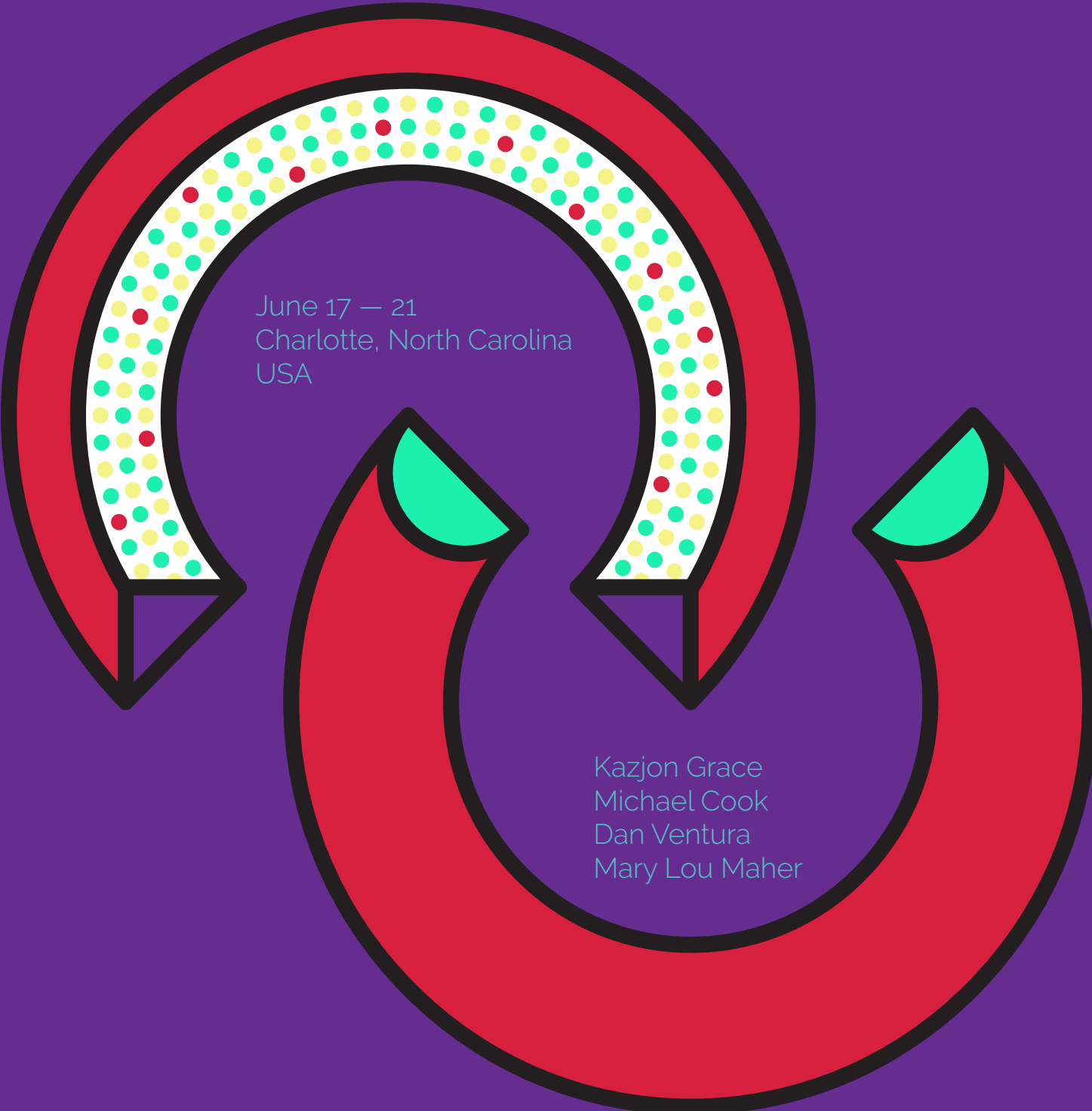


Proceedings of the 10th International Conference on Computational Creativity



June 17 – 21
Charlotte, North Carolina
USA

Kazjon Grace
Michael Cook
Dan Ventura
Mary Lou Maher



Proceedings of the Tenth International Conference on
Computational Creativity

ICCC 2019

Charlotte, North Carolina, USA | 17 - 21 June

Kazjon Grace, Michael Cook, Dan Ventura, Mary Lou
Maher (Editors)

Published by the Association for Computational Creativity
(ACC)



University of North Carolina at Charlotte
USA

<http://computationalcreativity.net/iccc2019/>

First published 2019

TITLE: Proceedings of the 10th International Conference on Computational Creativity

EDITORS: Kazjon Grace, Michael Cook, Dan Ventura, Mary Lou Maher

ISBN: 978-989-54160-1-1

Published by the Association for Computational Creativity (ACC)

Copyright notice: all contents of these proceedings by ACC, the Association for Computational Creativity, published under a Creative Commons Attribution (CC BY) license <https://creativecommons.org/licenses/by/4.0/>, which allows unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.



Attribution CC BY



Preface

This volume contains the papers presented at ICCC 2019, the 10th International Conference on Computational Creativity held in Charlotte, North Carolina, USA from June 17th - June 21st, 2019 (<http://computationalcreativity.net/iccc2019/>). The conference was hosted at the University of North Carolina at Charlotte.

Computational creativity is the art, science, philosophy and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviours that unbiased observers would deem to be creative. With artificial intelligence playing an increasingly important role in our work, our leisure and our social spaces, the themes covered by this field are increasingly relevant and important to society. The ICCC conference series, organized by the Association for Computational Creativity since 2010, is the only scientific conference that focuses on computational creativity alone and also covers all its aspects.

We received 57 paper submissions, in four categories:

1. Technical papers, posing and addressing hypotheses about aspects of creative behaviour in computational systems;
2. System and resource description papers, describing the building and deployment of a creative system or resource to produce artefacts of potential cultural value in one or more domains;
3. Study papers which appeal to broader areas of artificial intelligence; which appeal to studies of the field of computational creativity as a whole; or which draw on complementary fields such as psychology, philosophy, cognitive science, mathematics, humanities or the arts;
4. Position papers, presenting an opinion on some aspect of the culture of computational creativity research, including discussions of future directions, past triumphs or mistakes and issues of the day.

Each submission was reviewed by our program committee and then received a metareview from our senior program committee and program chairs, with additional discussion where required. Papers were accepted based on quality, academic rigour and relevance to one or more of the conference's four paper categories. The result is a diverse program that reflects the changing trends of artificial intelligence and the state of the art in computational creativity research.

The committee accepted 25 full papers for oral presentation and 12 papers for poster presentation. The three days of the ICCC 2019 scientific program consisted of a series of exciting sessions for oral presentations and a special session for posters and demos.

In addition to our main track, ICCC 2019 also hosted a late-breaking papers track for short reports on emerging and newer work, receiving 19 such submissions. Submissions to this track were also reviewed by our program committee, and 13 were accepted based on quality and relevance to the main conference themes. These late-breaking papers were presented as shorter talks, either alongside full papers or in their own dedicated sessions, mixing in the latest results from the community alongside the full papers reporting on established research.

ICCC 2019 continued the tradition established in earlier years of showcasing creative submissions. Creative submissions accepted for exhibition included interactive artworks, virtual environments, computational creative systems and commercial products produced using computational creativity. 11 submissions to this track were reviewed by two members

of a creative program committee, based on both the product or system and an extended abstract detailing the contribution. 10 submissions were selected and their extended abstracts have been included in these proceedings. The creative submissions were exhibited at ICCC 2019 in their own dedicated session, with lightning talks by the contributors to introduce their works in the main track.

This year's conference included several co-located events, including the return of our Doctoral Consortium, the 7th International Workshop on Musical Metacreation (MUME), and the first workshop on Computational Creativity and Deep Generative Design. MUME also organised a live concert as part of the conference program. The conference also hosted its first AI competition, providing a platform for the Generative Design in Minecraft Competition to report on results and announce the winners of the 2018/2019 competition.

As in past years, ICCC 2019 awarded a Best Paper Award and a Best Student Paper Award. To mark the tenth ICCC conference, we also introduced the Most Influential Paper Award, as a way to honour a paper from the first ICCC conference that has had the most impact on computational creativity in the years since.

We wish to thank our sponsor, the College of Computing and Informatics at UNC Charlotte. We thank the program committee and the senior program committee for their hard work in reviewing papers. We also thank all those involved in organising ICCC 2019, the ACC steering committee, and those involved in organising and supporting the workshops, tutorials and doctoral consortium.

We also wish to thank the Computational Creativity community for the amazing effort and energy that has allowed this conference to flourish and develop into its tenth year – thank you to those for whom this is their tenth year, supporting the community's development from its modest beginnings, and thank you to those for whom this is their first year, contributing to another decade of growth and new ideas for this field.

ICCC 2019 Organizing Committee

General Chair: **Mary Lou Maher**, University of North Carolina Charlotte

Program Co-Chair: **Kazjon Grace**, University of Sydney

Program Co-Chair: **Michael Cook**, Queen Mary University of London

Art Exhibition/Demos Co-Chair: **Rob Saunders**, University of Sydney

Art Exhibition/Demos Co-Chair: **Petra Gemeinboeck**, University of New South Wales

Workshop Co-Chair: **Ollie Bown**, University of New South Wales

Workshop Co-Chair: **Nick Davis**, University of North Carolina Charlotte

Doctoral Consortium Chair: **John Gero**, University of North Carolina Charlotte

Proceedings Chair: **Dan Ventura**, Brigham Young University

Media Co-Chair: **Stephen MacNeil**, University of North Carolina Charlotte

Media Co-Chair: **Jin Goog Kim**, University of North Carolina Charlotte

Student Volunteer Co-Chair: **Celine Latulipe**, University of North Carolina Charlotte

Student Volunteer Co-Chair: **Johanna Okerlund**, University of North Carolina Charlotte

Program Committee

Senior PC Members

Oliver Bown - University of New South Wales
Daniel Brown - University of Waterloo
Amílcar Cardoso - University of Coimbra
Simon Colton - Queen Mary University of London
John Gero - University of North Carolina Charlotte and George Mason University
Pablo Gervás - Universidad Complutense de Madrid
Ashok Goel - Georgia Institute of Technology
Anna Jordanous - University of Kent
Nada Lavrač - Jozef Stefan Institute
Carlos León - Universidad Complutense de Madrid
Francois Pachet - Spotify
Alison Pease - University of Dundee
Rafael Pérez y Pérez - Universidad Autónoma Metropolitana at Cuajimalpa
Graeme Ritchie - University of Aberdeen
Rob Saunders - University of Sydney
Julian Togelius - New York University
Hannu Toivonen - University of Helsinki
Tony Veale - University College Dublin
Dan Ventura - Brigham Young University
Geraint Wiggins - Vrije Universiteit Brussel and Queen Mary University of London

PC Members

Margareta Ackerman - Santa Clara University
Wendy Aguilar - Universidad Nacional Autónoma de México
Fabrizio Balducci - University of Bari
Gabriella A. B. Barros - New York University
Tarek Besold - Alpha Health AI Lab
Debarun Bhattacharjya - IBM
Josep Blat - Universitat Pompeu Fabra
Paul Bodily - Idaho State University
Liam Bray - University of Sydney
David C Brown - Worcester Polytechnic Institute
Remco Chang - Tufts University
João Correia - University of Coimbra
Nicholas Davis - University of North Carolina Charlotte
Pablo Delatorre - Universidad de Cádiz
Arne Eigenfeldt - Simon Fraser University
Heather Freeman - University of North Carolina Charlotte
Björn Gambäck - Norwegian University of Science and Technology
Andrés Gómez de Silva Garza - Instituto Tecnológico Autónomo de México
Hugo Gonçalo Oliveira - University of Coimbra
Christian Guckelsberger - Queen Mary University of London
Ivan Guerrero - Universidad Nacional Autónoma de México
Matthew Guzdial - Georgia Institute of Technology

Sarah Harmon - Bowdoin College
Raquel Hervás - Universidad Complutense de Madrid
Amy K Hoover - New Jersey Institute of Technology
Mikhail Jacob - Georgia Institute of Technology
Colin Johnson - University of Kent
Maximos Kaliakatsos-Papakostas - Aristotle University of Thessaloniki
Anna Kantosalu - University of Helsinki
Robert Keller - Harvey Mudd College
Oliver Kutz - Free University of Bozen-Bolzano
Antonios Liapis - University of Malta
Heather Ligler - Georgia Institute of Technology
Simo Linkola - University of Helsinki
Maria Teresa Llano - Goldsmiths, University of London
Phil Lopes - École Polytechnique Fédérale de Lausanne
Roisin Loughran - DkIT
Penousal Machado - University of Coimbra
Brian Magerko - Georgia Institute of Technology
Pedro Martins - University of Coimbra
Jon McCormack - Monash University
Stephen McGregor - Queen Mary University of London
David Meredith - Aalborg University
Diarmuid O'Donoghue - Maynooth University
Ana-Maria Olteteanu - Freie Universität Berlin
Allison Parrish - New York University
Philippe Pasquier - Simon Fraser University
Enric Plaza - Agencia Estatal Consejo Superior de Investigaciones Científicas
Senja Pollak - University of Ljubljana
Edward J. Powley - Falmouth University
Ana Rodrigues - University of Coimbra
Gillian Smith - Worcester Polytechnic University
Bob Sturm - KTH Stockholm
Anne Sullivan - Georgia Institute of Technology
Tapio Takala - Aalto University
Alan Tapscott - Universidad Complutense de Madrid
Kıvanç Tatar - Simon Fraser University
Tatsuo Unemi - Soka University
Lav Varshney - University of Illinois at Urbana-Champaign
Luis Fabricio Wanderley Goes - Pontificia Universidade Católica de Minas Gerais
Georgios N. Yannakakis - University of Malta
Martin Znidarsic - Jožef Stefan Institute

Table of Contents

Session 1: Interaction and Co-Creativity

Toward Digital Progymnasmata	1
<i>Kyle Booten</i>	
SpaceSheet: Navigating Conceptual Space with a Spreadsheet Interface	9
<i>Tom White and Bryan Loh</i>	
Deep Learning in a Computational Model for Conceptual Shifts in a Co-Creative Design System	17
<i>Pegah Karimi, Mary Lou Maher, Nicholas Davis and Kazjon Grace</i>	

Session 2: Evaluation and Appreciation

Read Me Like A Book: Lessons in Affective, Topical and Personalized Computational Creativity	25
<i>Tony Veale</i>	
Computational Analysis of Content in Fine Art Paintings	33
<i>Diana Kim, Jason Xu, Ahmed Elgammal and Marian Mazzone</i>	
Learning to Surprise: A Composer-Audience Architecture	41
<i>Razvan C. Bunescu and Oseremen O. Uduehi</i>	

Session 3: Posters

Theoretical Framework for Computational Story Blending: From a Cognitive System Perspective	49
<i>Taisuke Akimoto</i>	
Duets Ex Machina: On The Performative Aspects of “Double Acts” in Computational Creativity	57
<i>Tony Veale, Philipp Wicke and Thomas Mildner</i>	
Churnalist: Fictional Headline Generation for Context-appropriate Flavor Text .	65
<i>Judith van Stegeren and Mariët Theune</i>	
Analyzing Art Works: The Six Steps Methodology	73
<i>Luis Fernando Gutiérrez and Rafael Pérez y Pérez</i>	
“She Offered No Argument”: Constrained Probabilistic Modeling for Mnemonic Device Generation	81
<i>Paul M. Bodily, Porter Glines and Brandon Biggs</i>	
NONOTO: A Model-agnostic Web Interface for Interactive Music Composition by Inpainting	89
<i>Théis Bazin and Gaëtan Hadjeres</i>	
Neural Drum Machine : An Interactive System for Real-time Synthesis of Drum Sounds	92
<i>Cyran Aouameur, Philippe Esling and Gaëtan Hadjeres</i>	

Reshaping Design Search Spaces for Efficient Computational Design Optimization in Architecture	100
<i>Likai Wang, Patrick Janssen and Guohua Ji</i>	
The HR3 System for Automatic Code Generation in Creative Settings	108
<i>Simon Colton, Alison Pease, Michael Cook and Chunyang Chen</i>	
Mechanisms of Artistic Creativity in Deep Learning Neural Networks	116
<i>Lonce Wyse</i>	
Emolift: Elevator Conversations Based on Emotions	124
<i>Alejandro Oñate, Gonzalo Méndez and Pablo Gervás</i>	

Session 4: Co-creative Performance

MASSE: A System for Music Action Selection Through State Evaluation	132
<i>Prashanth Thattai Ravikumar and Lonce Wyse</i>	
Affordance-based Generation of Pretend Object Interaction Variants For Human-Computer Improvisational Theater	140
<i>Mikhail Jacob, Prabhav Chawla, Lauren Douglas, Ziming He, Jason Lee, Tanuja Sawant and Brian Magerko</i>	
Exploring Conditioning for Generative Music Systems with Human-Interpretable Controls	148
<i>Nicholas Meade, Nicholas Barreyre, Scott C. Lowe and Sageev Oore</i>	

Session 5: Framing and Critique

Framing In Computational Creativity - A Survey And Taxonomy	156
<i>Michael Cook, Simon Colton, Alison Pease and Maria Teresa Llano</i>	
Summaries Can Frame—But No Effect on Creativity	164
<i>Leonid Berov</i>	
Critique as Creativity: Towards Developing Computational Commentators on Creative Works	172
<i>Douglas H. Fisher and Haerin Shin</i>	

Session 6: Creative Machine Learning

Combinets: Creativity via Recombination of Neural Networks	180
<i>Matthew Guzdial and Mark Riedl</i>	
Trick or TReAT : Thematic Reinforcement for Artistic Typography	188
<i>Purva Tendulkar, Kalpesh Krishna, Ramprasaath R. Selvaraju and Devi Parikh</i>	
Beyond Imitation: Generative and Variational Choreography via Machine Learning	196
<i>Mariel Pettee, Chase Shimmin, Douglas Duhaime and Ilya Vidrin</i>	

Session 7: Narrative and Poetry

TwitSong 3.0: Towards Semantic Revisions in Computational Poetry	212
<i>Carolyn Lamb and Daniel G. Brown</i>	
Evolving the INES Story Generation System: From Single to Multiple Plot Lines	220
<i>Eugenio Concepción, Pablo Gervás and Gonzalo Méndez</i>	
Generating a Search Space of Typical Narrative Plots	228
<i>Pablo Gervás</i>	

Session 8: Metapapers

Field Work in Computational Creativity	236
<i>Margareta Ackerman and Rafael Pérez y Pérez</i>	
The Draw-A-Computational-Creativity-Researcher Test (DACCRT): Exploring Stereotypic Images and Descriptions of Computational Creativity	243
<i>Sarah Harmon and Katie McDonough</i>	
The Importance of Applying Computational Creativity to Scientific and Mathematical Domains	250
<i>Alison Pease, Simon Colton, Chris Warburton, Athanasios Nathanail, Irina Preda, Daniel Arnold, Daniel Winterstein and Mike Cook</i>	

Session 9: Humour and Colourful Language

No Time Like the Present: Methods for Generating Colourful and Factual Multilingual News Headlines	258
<i>Khalid Alnajjar, Leo Leppänen and Hannu Toivonen</i>	
Modelling the Socialization of Creative Agents in a Master - Apprentice Setting: The Case of Movie Title Puns	266
<i>Mika Hämäläinen and Khalid Alnajjar</i>	
Towards a General Framework for Humor Generation from Rated Examples ...	274
<i>Thomas Winters, Vincent Nys and Daniel De Schreye</i>	

Late Breaking Papers

MindMusic: Brain-Controlled Musical Improvisation	282
<i>Rachel Goldstein, Andy Vainauskas, Margareta Ackerman and Robert M. Keller</i>	
Situated Novelty in Computational Creativity Studies	286
<i>Marija Majda Perišić, Mario Štorga and John Gero</i>	

Going into Greater Depth in the Quest for Hidden Frames	291
<i>João Gonçalves, Aaron Bembenek, Pedro Martins and Amílcar Cardoso</i>	
Assessing Usefulness of a Visual Blending System: "Pictionary Has Used Image-making New Meaning Logic for Decades. We Don't Need a Computational Platform to Explore the Blending Phenomena", Do We?	296
<i>João Miguel Cunha, Sérgio Rebelo, Pedro Martins and Penousal Machado</i>	
A Dynamic Approach for the Generation of Perceptual Associations	301
<i>Ana Rodrigues, Amílcar Cardoso and Penousal Machado</i>	
Performing Structured Improvisations with Pre-trained Deep Learning Models	306
<i>Pablo Samuel Castro</i>	
Generative Design in Minecraft: Chronicle Challenge	311
<i>Christoph Salge, Christian Guckelsberger, Michael Cerny Green, Rodrigo Canaan and Julian Togelius</i>	
Shallow Art: Art Extension Through Simple Machine Learning	316
<i>Kyle Robinson and Dan Brown</i>	
Engagement-Reflection in Software Construction	321
<i>Quinten Rosseel and Geraint A. Wiggins</i>	
Modeling Knowledge, Expression, and Aesthetics via Sensory Grounding	326
<i>Brad Spendlove and Dan Ventura</i>	
Exploring the Notion of Self in Creative Self-Expression	331
<i>Vikram Jamwal</i>	
Region Radio: An AI that Finds and Tells Stories about Places	336
<i>Douglas H. Fisher, Emily Markert, Abigail Roberts and Kamala Varma</i>	
Opportunities for Computational Creativity in a Therapeutic Context	341
<i>Lee Cheatley, Wendy Moncur and Alison Pease</i>	

Art and Demo Abstracts

WanderGAN	346
<i>Eyal Gruss</i>	
Pictures of J** Girls in Synthesis	348
<i>Eyal Gruss, Ayelet Sapirshstein and Vered Heruti</i>	
Algorithm as Determinant of Form in Music	350
<i>Halley Young</i>	
Hidden Worlds	352
<i>J. Rosenbaum</i>	
Fragile Pulse: A Meditation App	354
<i>Kyle Booten</i>	
Emotion-driven Creative Music Composition in brAInMelody®	356
<i>Masayuki Numao</i>	

Creative Sketching Partner: A Co-Creative Sketching Tool to Inspire Design Creativity	358
<i>Nicholas Davis, Safat Siddiqui, Pegah Karimi, Mary Lou Maher and Kazjon Grace</i>	
Two Fragrances Designed for Brazilian Millennials	360
<i>David Apel, Sophie Bensamou, Veronique Ferval, Jing Fu, Richard Goodwin, Christian Harris, Andrea Lombardi, Robin Lougee, Joana Maria, Richard Segal and Tenzin Yeshi</i>	
Computational-Creativity Enabled One Pan Seasonings for Retail Sale	362
<i>Jing Fu, Richard Goodwin, Christian Harris, Kimberly Lang, Robin Lougee, CJ McLane, Joana Maria, Jim Martin, Richard Segal, Tenzin Yeshi</i>	
ChordAL: A Chord-Based Approach for Music Generation using Bi-LSTMs ...	364
<i>Tan Hao Hao</i>	

Toward Digital *Progymnasmata*

Kyle Booten

Neukom Institute for Computational Science
Dartmouth College
Hanover, NH 03755 USA
kyle.p.booten@dartmouth.edu

Abstract

The classical arts of rhetoric described intricate training methodologies for making the writer linguistically flexible and able to avoid stylistic vices. Inspired by the ancient *progymnasmata*, this paper presents Progym, an interactive writing system designed to notice when writers resort to expected language and encourage them to avoid these linguistic elements. Two versions of the system are presented. The first discourages writers from using words that, within a large corpus, are often used to describe a target word. The second discourages writers from using syntactic patterns found in a small corpus. In user studies, Progym did indeed push writers away from these features, though the different versions led to different styles of revision.

Introduction

From its roots in antiquity through its second zenith during the Early Modern period, the arts of rhetoric provided learners with exercises designed to hone their use of language. While much of rhetorical practice was grounded in the imitation of received forms and authors, this does not mean that it did not also foster creativity. Among the ancient Greek *progymnasmata* (a set of preliminary rhetorical exercises) were *ekphrasis*—the description of an object or artwork with a vivid attention to detail; another such exercise was *paraphrase*, the reiteration of a statement with different syntax (Kennedy 2003). The point of these and other *progymnasmata* was not that they themselves produced full or complete texts; rather they were a kind of “gymnastic training for the mind...shaping it for certain activities just as athletics shaped the body” (Webb 2001). A similar spirit of athleticism can be seen much later in Erasmus’ treatise on rhetorical education *De Copia* (1512 1978), which recommends various techniques for “diversifying” one’s speech or writing and avoiding monotony. Demonstrating a rhetorical exercise meant to promote linguistic flexibility, Erasmus’s text offers over a hundred and fifty distinct variations on a simple phrase, the Latin equivalent of “Your letter has delighted me very much.”

This paper documents the design of a system that provides computational feedback as a form of rhetorical training in the context of creative writing tasks. Inspired by the gymnastic notion of language found in the rhetorical tradi-

tion, and especially by Erasmus’ example of forcing oneself into linguistic “copiousness” or flexibility, this system is designed to encourage creativity by steering writers away from particularly common and expected words and syntactic patterns. Like the classical *progymnasmata*, the system is not primarily designed to produce complete or sufficient texts. Rather it is conceived of as a training tool designed to encourage linguistic flexibility. On a technical level, this paper describes techniques for gathering overly-frequent linguistic phenomena using text mining. This paper documents the design of two different versions of this progymnastic system. Results from user studies document the impacts the system’s different types of feedback had on the ways that writers used language.

Related Work

Computational Writing Assistants

Within the field of computational creativity, researchers have developed systems that assist humans in the production of creative writing. Some of these computational systems function as collaborators. Say Anything (Swanson and Gordon 2008) functions as a kind of creative Information Retrieval system for narrative composition, returning a sentence from a large collection of texts that is most similar to the human writer’s. Inspired by this system, Creative Help (Roemmele and Gordon 2015) uses similar techniques to match human input with a sentence from a large corpus, although it allows writers to more flexibly control how they deploy these sentences. The system approaches interactive storytelling as an Information Retrieval task, with the algorithmic writer returning a sentence from a large collection of sentences that is the most similar to the user’s. More recent research from Roemmele (2016) has explored the use of the predictive models of neural networks as an improvement upon traditional techniques of Information Retrieval for offering suggestions to writers as they write. Manjavacas et al. (2017) also used a language model to provide continuations of a human writer’s text.

Creative computation research on writing assistants has also drawn on research within the field on the generation of literary texts. For instance, “Co-PoeTryMe” (Oliveira, Mendes, and Boavida 2017) is an interactive version of PoetTryMe (Oliveira 2012), a system for generating poetry

in multiple languages using a combination of networks of semantically-related words and a variety of syntactic and formal constraints, including rhyme and number of syllables. Co-PoeTryMe makes this poetry generation tool interactive by providing an interface for specifying the parameters of the generator and for iterative generating and editing words and lines. Inkwell (Gabriel, Chen, and Nichols 2015) is another system that is both a poetry generator and a poetry-writing assistant. As an assistant, it combines a wide variety of individual functions, such as mimicking a writer’s personality and style.

Creative assistants for writing may also provide something like “inspiration” rather than engage in full-fledged collaboration. Gonçalves et al. (2017) demonstrated a system that uses what they call “subliminal priming” to provide writers with feedback to help them get over writer’s block. The Poetry Machine (Kantosalo et al. 2014; Kantosalo, Toivanen, and Toivonen 2015), another repurposing of a poetry generation system (Toivanen et al. 2012), offers the writer initial “fragments” of poetry as a way to help them overcome the difficulty of starting the writing process. Indurkha (2016) used a similar approach, providing writers (in this case, children) with a combination of related and unrelated words in order to both scaffold the production of a narrative and spur creativity. Researchers have also used crowdsourced images to stimulate creativity and mental well-being during a creative writing task (Gonçalves and Campos 2018).

Progym does not position itself as a “collaborator.” Neither does it supply the writer with fragmentary suggestions with the goal that, by integrating them, the writer may make a text more compelling (or merely overcome some of the psychological barriers of writing, such as writers block). Neither does it aim to make the writer feel better while writing. Rather it offers explicitly negative feedback to direct writers to be more creative. In this sense, it is a kind of “coach” (Lubart 2005) as well as a kind of “audience” (Riedl and O’Neill 2009), albeit an opinionated and in fact critical one. The main contribution of this paper is to explore how a system can ask a writer to avoid certain kinds of uncreativity.

Mining Semantic Relations

One version of the Progym system is based on semantic relations between words mined from a large corpus of texts. The notion of mining texts for semantic relations was described by Hearst (1992). Related techniques have been used to mine semantic relationships between words as a way to generate poetry (Toivanen et al. 2012; Veale 2013) and metaphor (Veale and Hao 2007). Veale and Hao’s “Jigsaw Bard” (2011) turns semantic relations mined from the web into “a creative thesaurus” of metaphors—in a way, another kind of creative writing assistant.

A main goal of this paper is to take this familiar approach to extracting semantic connections from large corpora and use it in the context of a writing assistant that explicitly wants the user to avoid these statistically-predictable semantic connections between words.

moon (adj)	full new bright young pale old white great high waning
moon (verb)	shine shin ² hang set sink arise shed light climb cast
moon (noun)	light ray face surface beam disc or- bit disk revolution distance
tree (adj)	old great tall large big young green small hollow beautiful
tree (verb)	spread bear wave blossom surround bend bud live hang overhang
tree (noun)	shade branch life root shadow side heart leaf head crown
queen (adj)	young little great beautiful fair good new old dead poor
queen (verb)	send sit die reign wear speak live think smile hear
queen (noun)	room chamber apartment death hand presence command eye taste heart
wolf (adj)	hungry gray old big grey great young large fierce dead
wolf (verb)	howl eat prowl devour creep leap kill roam attack catch
wolf (noun)	head mouth den skin tooth howl fang tongue eye tail

Table 1: Most Frequently Related Words (Lemmatized) Extracted from Project Gutenberg Text

Progym V.1: Avoiding Expected Words

The sun is bright. The sun shines. The sun has beams. Compare these plausible assertions to the following: *The sun is dim. The sun blinks. The sun has banners.*

The first version of the Progym systems aims to steer writers away from the former—that is, from plausible but common descriptions of a topic noun and toward less common ones.¹

Finding Common Words Common relationships between words were mined from the a selection of the Project Gutenberg corpus using the SpaCy dependency parser (Honnibal and Johnson 2015), which represents any input sentence as a directional graph of syntactic as well as semantic relationships between words. Using this parser, the following relationships were extracted:

-Adjectival Relations For any noun, the system extracted adjectives that were the child of that noun via an `adjmod` (adjectival modifier) dependency relation. For instance, from the sentence “The old man is weary” it would extract

¹This can be thought of as encouraging “creativity” in the broad sense that deviation from a statistically-common pattern amounts to a subversion of a “priming” (Hoey 2007).

²Ostensibly an artifact of inconsistencies in the lemmatization of verb forms of “shine.”

(man,old) and (man,weary), using the lemmatized version of the noun.

-Possessive Relations For any noun, the system found all nouns that were the child of this noun via a `poss` (possession modifier) dependency relation. For instance, from the sentence “The dog’s fur is golden” it would extract (dog, fur), using the lemmatized version of the noun.

-Verb Relations For each noun, the system found the verb that was the parent of this noun via a `nsubj` (nominal subject) relation. In addition, for each noun, the system found the present participle (tagged `VBG`) that was the child of the noun via an `adjmod` relation. For instance, from the sentence “The howling wolf chased me” it would extract (wolf,howl) and (wolf,chase), using the lemmatized version of nouns and verbs.

Using these techniques, fragments were mined from each of 14,928 English-language texts from Project Gutenberg; this is a collection of open source texts of a mostly literary nature, and so it was both convenient and, since I wanted to mine relations that appear in literary language, befitting of the task. Mining fragments was limited to the first 100,000 characters of each text, a limit imposed to ensure a reasonable compute time. Each word and each pair was further verified to be a valid word with the correct part of speech using WordNet (Miller 1995). To deal with the fact that certain uses of words may be idiosyncratic to a particular author, each text within the selection of the Gutenberg corpus was only able to contribute a specific relation between a noun and another word at most once. Using these criteria, an average of 322 Adjectival Relations were discovered for 27,444 nouns, an average of 176 Verb Relations were found for 26,443 nouns, and an average of 45 possessive relations were found for 6,729 nouns. Table 1 shows some of the top nouns, adjectives, and verbs found through these relations for several target nouns.

For each noun, a threshold was set either at 3 or at the number of occurrences of the pair at the 90th percentile of all observed relations of that specific type, whichever was higher. This was done to deal with rare nouns or nouns with few relations of a specific type, especially since even relatively few Possessive Relations were extracted overall. For Verb Relations and Adjectival Relations, certain very common words (such as “is” and “such”) were treated as stop words and excluded. This process produced, for each noun, a list of Boring Words—Boring Verbs, Boring Adjectives, and Boring Nouns.

Interface

The Progym system is deployed as a web-based interface designed specifically for the user study (see Figure 1). The interface itself is straightforward and minimalistic, presenting the user with a series of ten text input areas. It is intended to be used in the context of an ekphrastic task in which a user must write ten sentences about a specific noun.

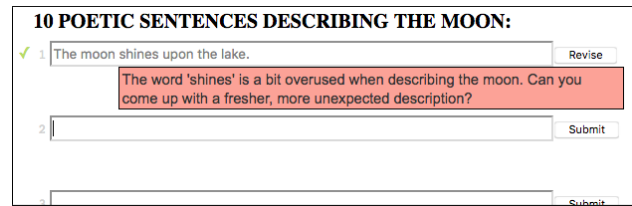


Figure 1: Progym’s Interface

Each time the user “submits” a sentence, the system part-of-speech tags the sentence and checks its adjectives, lemmatized verbs, and lemmatized nouns against that noun’s Boring Words. If a Boring Word is detected, Progym presents the user with a message asking them to revise it—e.g. “The words ‘fluffy’ and ‘white’ are a bit overused when describing a cloud. Can you come up with a fresher, more unexpected description?” If the input sentence contains, for instance, both one of the target noun’s Boring Adjective and one of its Boring Verbs, it randomly focuses on one part-of-speech, and at most two different words of this part-of-speech. Users can then revise and resubmit their sentences, once again triggering the system’s evaluation so that for the critical comment to disappear all Boring Words must be purged from the sentence.

User Study 1

For the purposes of the user study, Amazon Mturk crowdworkers were asked to write ten “poetic” sentences, each describing a different aspect of *the moon* or *a tree*.³ These words were chosen as they are both relatively high-frequency nouns with correspondingly ample numbers of Boring Words (for “moon,” 29 Boring Nouns, 26 Boring Verbs, and 53 Boring Adjectives; for “tree,” 15 Boring Nouns, 56 Boring Verbs, and 111 Boring Adjectives). In addition, from Percy Bysshe Shelley’s “To the Moon” to Coleridge’s “This Lime-Tree Bower My Prison,” both topics have a long history as objects of ekphrastic description. Workers were either given feedback by Progym ($n = 33$ for moon, $n = 42$ for tree) or not ($n = 44$ for moon, $n = 47$ for tree).

Use of Expected Words

Progym’s functions for identifying the use of Boring Words were repurposed for the analysis of the sentences written by the Mturk participants under the four conditions, Tree-Assisted (by Progym’s suggestions), Moon-Assisted, Tree-Unassisted, Moon-Unassisted.

Participants could revise a sentence multiple times, and the system recorded each revision to each of the participant’s ten sentences. As these writers revised according to Progym’s feedback in the assisted conditions, they lessened the number of Boring Words in their texts. Looking at the earliest version of sentences, Moon-Assisted poems had an

³The Github library `quickstart-mturk` was adapted with the permission of its author, user `akuznets0v`.

average of 7.31 Boring Words ($SD = 3.17$); looking at the most recent (i.e. “final”) version of sentences, they had an average of 3.90 ($SD = 3.60$), a statistically significant difference according to a two-tailed t-test, $t(82) = 7.10, p < .001$. Looking at the earliest version of sentences, Tree-Assisted poems had an average of 6.03 Boring Words ($SD = 3.49$), while the most recently-revised versions had an average of 3.21 ($SD = 4.26$), also a statistically significant difference according to a two-tailed t-test, $t(64) = 6.81, p < .001$.⁴

Likewise, writers who had the assistance of the system ended up with sentences with fewer Boring Words overall than the control (unassisted) condition. The ten-sentence exercises of Tree-Unassisted and Moon-Unassisted conditions had an average of 10.91 ($SD = 4.76$) and 6.93 ($SD = 3.16$) Boring Words, respectively. By contrast, the ten-sentence exercises of Tree-Assisted and Moon-Assisted conditions had an average of 3.90 ($SD = 3.60$) and 3.21 ($SD = 4.26$), respectively. This differences between assisted and unassisted conditions were statistically significant for tree conditions according to a two-tailed t-test, $t(87) = 7.68, p < .001$, and for moon conditions, $t(75) = 4.35, p < .001$.

Analyzing the unassisted conditions provide a way to check that Progym’s sense of what counts as a Boring Noun for a particular word is sensible. Compared to the above-stated average of 10.91 Boring Words for the noun “tree” in the Tree-Unassisted condition, an average of 3.19 ($SD = 2.32$) Boring Words for the noun “moon” (i.e the “incorrect” words) were found, a statistically significant difference, $t(92) = 9.90, p < .001$. Likewise, compared to the above-stated average of 6.93 Boring Words for the noun “moon” in the Moon-Unassisted condition, an average of 4.09 ($SD = 2.50$) Boring Words for the noun “tree” were found, a statistically significant difference, $t(86) = 4.62, p < .001$. In other words, Progym’s noun-specific lists of Boring Words mined from Project Gutenberg texts were predictive of the ways that participants in the user study wrote about these two particular nouns.

Qualities of Revision

To analyze the ways that participants wrote when confronted with Progym’s criticism, for all sequential pairs of revisions $((s_0, s_1), (s_1, s_2) \dots)$ the Levenshtein distance in terms of tokens was calculated, with one outlier removed.⁵ Figure 2 shows the distribution of the frequency of lengths of revisions produced by users in the Inspiration-Assisted condition. There were 323 revisions total, with an average of 4.31

⁴Analysis of the data revealed that participants did not always heed the study-task’s exhortation that they write ten sentences in ten different text boxes; sometimes they wrote more than one sentence in a text box. To control for the length of users’ writing, calculations in section result from analysis of the first actual sentence of each user’s ten input texts, as determined by the SpaCy parser’s sentence tokenization.

⁵Several of these pairs contained edit distances much greater than the average. Upon closer inspection, these can be explained by the fact that entire poems by poets such as Robert Frost were submitted, with the previous or sequential “revision” of that line being much shorter and in fact unrelated. Revisions of an edit distance greater than or equal to 150 were excluded.

blue	→	azure
face	→	visage
changing	→	periodic
limbs	→	appendices
surface	→	topography
bends	→	swoops
beautiful	→	sightly
beautiful	→	spellbinding

Table 2: Example Revisions Made Toward Rarer Word

revisions per participant ($SD = 3.58$). The majority of revisions were of an edit distance of 1.

What were the nature of these one-token changes? By encouraging writers to avoid common words, the system also pushed writers toward greater linguistic diversity. Those revisions were gathered in which the user’s original sentence and first revision of this sentence were equal in number of tokens but differed by exactly one token—i.e. in which one token (w_0) was “replaced” by another (w_1). Out of the 108 w_0 tokens, there were only 64 unique ones. By contrast, there were 102 unique w_1 tokens, a statistically significant difference according to a chi-squared test, $\chi^2(1) = 35.62, p < .001$. In essence, the collection of “revised” words was more varied than the collection of “unrevised” words.

It was hypothesized that pressure from Progym may encourage writers to eschew common words, replacing them with rare ones. Google Ngram Viewer⁶ provides a way of roughly testing whether one word is more common than the other. For each pair of sequential revisions that were equal in number of tokens but differed by one word, Ngram Viewer was used to check whether the word in the first sentence, w_n , or the word that replaced it, w_{n+1} , was the more frequent.⁷ Out of 167 of such comparisons, w_{n+1} was the rarer word in 116 (69%), a statistically significant difference according to a chi-squared test, $\chi^2(1) = 25.30, p < .001$. The difference was a bit more extreme looking only at those revisions in which the first version of a sentence was equal in number of tokens to its “final” version but differed by one word; of these (w_{first}, w_{last}) pairs, w_{last} was the rarer word 76% of the time (65 out of 87), a statistically significant difference, $\chi^2(1) = 21.25, p < .001$. This suggests that Progym inspired participants to use less-frequent words. Table 2 shows a sample of the single word revisions in which a word was substituted by a rarer one.

Progym V.2: Beyond the Word

The second version of Progym differs from the first in two respects. First, rather than focus on individual words, it encourages the users to turn away from too-common syntax. Second, rather than compare the writer to specific relations mined and distilled from a very large number of texts, it compares the writer to a relatively small number of exam-

⁶<https://books.google.com/ngrams>

⁷Datapoints for the year 2000, the default most recent year, were compared. Automatic spelling correction was applied using the PyEnchant library.

you can do	←	You can do anything you want to do, you just need to push yourself sometimes to get them done.
VB your NN	←	Focus your energy and you can make leaps and bounds
RB VB up	←	NEVER GIVE UP
you are JJ	←	You are smart and intelligent.
do n't VB	←	Don't give up. You'll be glad you didn't.
if you VBP	←	If you stop now, all the work you've put in thus far will have been for nothing.

Table 3: Rhetorical Stubs Used by Progym V.2 (Most Frequent in Corpus), with Examples

ples. Using the same interface as before, “inspiring” sentences were gathered from Amazon Mturk crowdworkers. These workers were told: “Imagine that you are writing for somebody who needs your words to help them accomplish a difficult task or overcome some adversity.” In all, ten sentences each from 49 crowdworkers were collected.

These sentences became a small corpus of examples to which Progym would compare any new inspiring sentence, testing its novelty against them. The goal of this version of Progym is to push users away from the one-word edits typical of interactions with V.1 by focusing on longer syntactic units rather than individual words. It does so by comparing the syntax writers use to begin their inspiring sentences.

For each sentence in the example sentences, at most the first three tokens were either represented as this token’s part-of-speech tag or, if this token was in a list of stop words⁸, the token itself. For instance, the sentence “Focus your energy and you can make leaps and bounds” is represented as (*VB, your, NN*). Figure 3 shows the most frequent stubs in this small corpus with examples. This technique of building abstract—but not totally unlexicalized—representations of text is inspired by the “stretchy patterns” described by Gianfortoni, Adamson, and Rosé (2011). Since the goal of this exercise was to target patterns that may be overused in specifically inspiring sentences (rather than sentences in general), the top 20 most frequently used of such patterns in an excerpt of the Gutenberg Corpus were excluded, leaving 308 in all (see Table 2).

Progym V.2 asks users to generate inspiring sentences, testing how they begin against these banned “Rhetorical Stubs” found in the previously-gathered example sentences. When there is a match between the writer’s sentence and one of the examples, Progym once again provides feedback like this: “The phrase ‘You are ready’ reminds me of other inspiring messages, like ‘You are amazing and nothing can stop you.’ Could you try making yours a little more creative?” Rhetorical Stubs are meant to strike a balance between the semantic openness of merely a part-of-speech take sequence and the specificity of the sequence of tokens themselves, drawing attention away from the choices of words toward the underlying structure of the sentence. In other words, while one may substitute the participle “running” with any number of words (e.g. “sprinting,” “hustling,” and “galloping”), one may not so easily replace a closed-class word such as “you.” The design choice of the “Rhetori-

⁸Here the standard list in the Natural Language Toolkit (Bird, Klein, and Loper 2009) was used and supplemented with tokens to accommodate how the SpaCy parser tokenizes contractions (e.g. “ll”).

cal Stub” was made to stimulate revisions unlike those one-word revisions users made when interacting with V.1.

User Study 2

Amazon Mturk crowdworkers were tasked with writing ten inspiring sentences, either assisted by Progym ($n = 35$) or unassisted ($n = 38$).

Use of Rhetorical Stubs

Progym’s function for identifying the use of banned Rhetorical Stubs was re-purposed for the analysis of the sentences written by the Mturk participants under two conditions, Inspiration-Assisted and Inspiration-Unassisted.

As writers revised according to Progym’s V.2’s feedback in the assisted conditions, they lessened the number of banned Rhetorical Stubs in their texts. Looking at the earliest version of sentences (i.e. before any revision based on Progym’s suggestions), the poems of the assisted condition had an average of 3.69 banned Rhetorical Stubs ($SD = 1.74$); looking at the most recent (i.e. “final”) version of sentences, they had an average of 1.26 ($SD = 1.87$), a statistically significant difference according to a two-tailed t-test, $t(68) = 5.55, p < .001$. Participants writing with assistance of Progym V.2 ended up with sentences with fewer banned Rhetorical Stubs compared to the control (unassisted) condition. The ten-sentence exercises of Inspiration-Unassisted and Inspiration-Assisted had an average of 4.47 ($SD = 2.02$) and 1.26 ($SD = 1.87$), respectively. This difference was statistically significant according to a two-tailed t-test, $t(71) = 6.94, p < .001$.

To test whether the Progym V.2’s small number of Rhetorical Stubs were, as one would expect, a reasonable “training set,” a comparison was made between the number of banned Rhetorical Stubs in the Inspiration-Unassisted condition and (as an example of non-inspirational sentences written under similar experimental conditions) the Unassisted Tree and Moon conditions from the previous user test. One would expect the banned Rhetorical Stubs generated from the example sentences have better “coverage” of additional inspiring sentences than uninspiring ones. (Otherwise, those Rhetorical Stubs may simply be characteristic of sentences generally produced by Mturk workers, no matter what the rhetorical or expressive purpose.) Indeed, compared to an average of 4.47 of those Rhetorical Stubs found in the Inspiration-Unassisted condition, there were an average of 1.41 ($SD = 1.52$) found in the collection of Moon and Tree-Unassisted conditions, a statistically significant difference according to a two-tailed t-test, $t(127) = 9.33, p < .001$. In this case, even a small number of example sentences were predictive of the

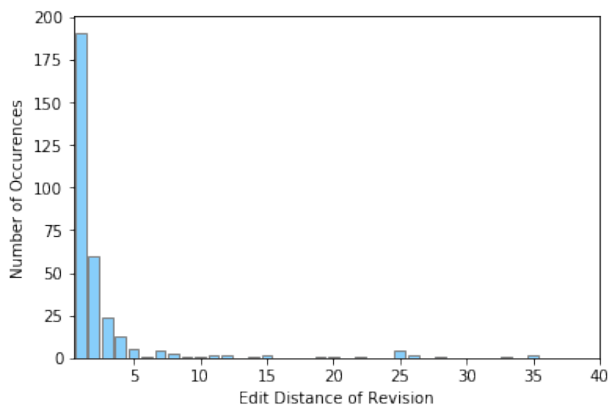


Figure 2: Revisions with Progym V.1 (Tree and Moon)

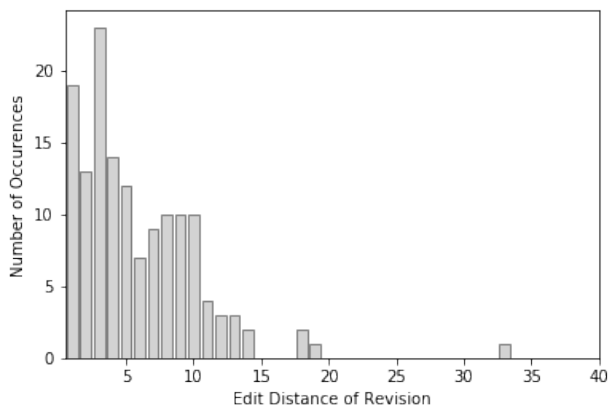


Figure 3: Revisions with Progym V.2 (Inspiring Sentences)

kinds of Rhetorical Stubs that would be written in other examples of inspiring sentences.

Coverage Due to Delexicalization Making Rhetorical Stubs is more computationally complex than simply using, for instance, the first three tokens from the example inspiring sentences in the training data (what might be called “Simple Stubs” [n = 402]). However, because they are more “general” (i.e. mostly delexicalized), the Rhetorical Stubs made out of these had better coverage over the data. Compared to a per-poem average of 4.47 of those Rhetorical Stubs found in the Inspiration-Unassisted condition, there were only 2.47 Simple Stubs ($SD = 2.02$), a statistically significant difference according to a two-tailed t-test, $t(74) = 4.25, p < .001$. Delexicalizing was thus an effective way to “stretch” data.

Qualities of Revision

Once again, all sequential pairs of revisions $((s_0, s_1), (s_1, s_2) \dots)$ were analyzed for the edit-distance (in terms of tokens) between the two. Figure 3 shows the distribution of the frequency of lengths of revisions

produced by users in the Inspiration-Assisted condition (as in the calculations for V.1, with several outliers removed). Comparing this to the frequency of lengths of revisions produced by users interaction with V.1, which were mostly a single token in length, these revisions show a tendency toward revisions of multiple tokens. There were 143 revisions total, with an average of 4.09 per participant ($SD = 2.89$). The average edit distance of the revisions created by participants using V.1 was 3.00 ($SD = 5.35$, median = 1), while the average edit distance of the revisions created by participants using V.2 was 5.87 ($SD = 4.52$, median = 5), a statistically significant difference according to a two-tailed t-test, $t(464) = 5.57, p < .001$. Moreover, as can be seen by comparing and Figure 2 and Figure 3, the lengths of revision completed with V.2 are more diverse. While for V.2 the top revision length was indeed 3 (reflecting the fact that the prompt drew attention to a Rhetorical Stub made from three tokens), revisions were more likely to be other lengths than revisions made with V.1 were likely to be lengths other than 1. This diversity can be described statistically: the entropy of the revisions performed with V.2 ($n = 143$) was 2.55 bits. By contrast, the entropy of a random sample of the same number of revisions performed with V.1 was 1.34 bits, this lower entropy signalling less diversity in the revision lengths.

Like Progym V.1, V.2 seemed to encourage linguistic diversity. For each sequential pair of revisions, the first or “unrevised” Rhetorical Stub (rs_0) and the subsequent revision (rs_1) were gathered. Out of the 93 rs_0 patterns, there were only 56 unique ones. By contrast, there were 85 unique rs_1 patterns, a statistically significant difference according to a chi-squared test, $\chi^2(1) = 22.98, p < .001$. The collection of “revised” Rhetorical Stubs was more diverse than the collection of “unrevised” ones. By putting pressure on writers to avoid certain common Rhetorical Stubs, Progym nudged them toward linguistic variation.

There was no evidence that revisions using V.2 led to an increase in the rarity of words within a text, though the consideration of this was limited by the small number of (w_n, w_{n+1}) word pairs ($n = 15$). Of these, w_{n+1} was the rarer word in 9 of them—not a statistically significant difference, $\chi^2(1) = 0.60, p > .05$.

Another Pattern of Revision For all sequences of revision of at least length 2 (i.e. in which the writer revised a sentence once and then revised again, $n = 37$), were gathered, and the first, second, and last (final) versions of these sentences were compared. In 6 of these, the writer first changed the sentence such that one of the first three tokens was different but it still matched the same “forbidden” Rhetorical Stub as the original sentence before ultimately revising the sentence more dramatically in a way that manifested a different Rhetorical Stub. For instance:

- You are *enough* just as you are.
- You are *perfect* just as you are.
- your attitude determines your direction [sic]

In such cases, it seems that the flexibility of the Rhetorical Stub has pushed the writer beyond simply swapping out a word with another related word of the same part of speech.

Discussion

Two versions of Progym were tested. Each version effectively steered writers away from certain linguistic elements that the system desired them to avoid. The two versions of Progym led to different styles of revision: participants writing with V.1 produced mostly single-word changes, shifting a common word to a rare one. Those writing with V.2 engaged in more extensive revision in terms of the number of tokens changed. Both small and large corpora of examples were useful for creating a background of “expected” language against which writers were asked to depart and encouraging linguistic diversity. This study was limited in the sense that it focused on only on several conditions (the Tree vs. Moon conditions, and the Inspiring conditions). Future research could explore a wider set of each of these.

Conclusions

This paper’s title begins with the word “toward” in order to make clear that its goal is to test the validity of a path. The main conclusion to be drawn from it is that even relatively simple techniques for predicting what users will write can be used to steer them away from these predictable moves and encourage linguistic diversity as well as different techniques of revision. One may imagine, further down the path, a wide variety of digital *progymnasmata* that would train writers to spurn mundane formulations or vary their styles.

Future versions of digital progymnastic systems could no doubt make use of more complex computation to determine whether a writer is veering into some too-common pattern or formulation. For instance, one might use a more complex statistical approach to identify clichés (Smith, Zee, and Uitdenbogerd 2012) or make use of statistical models of character types (Bamman, O’Connor, and Smith 2013) to detect when users are falling into common tropes. Likewise, a more complicated interface could allow the writer to have more control over the system—for instance, by specifying that they want to practice avoiding familiar syntactic construction or words, by adjusting the level of difficulty, or by specifying certain discourses that they want to depart from (e.g. the syntactic constructions of Romantic poetry in particular). A larger problem is how to address the fact that writers may use “boring” words or syntactic constructions in nonetheless interesting ways. For instance, while to write that “the moon is white,” may be overly expected, to write that “the moon is white like your Toyota Prius” may seem less so. Likewise, a sentence may use expected words organized in rhetorically powerful ways; a more complete system would keep an eye out for figures such as *anaphora* or *chiasmus* (Dubremetz and Nivre 2015). However, just as simple systems of text generations can serve as a baseline for more complex systems (Montfort and Fedorova 2012), it is useful to explore a pair of relatively straightforward techniques for steering writers away from “predictable” language to which more complex ones may be later compared.

This paper has focused on the way that Progym “mediated” (Vygotsky 1980) writers’ writing process. However, while crowdsourcing interactions with the system allowed for statistical analysis of these interactions, this research

could be complemented by a more naturalistic and qualitative study of student or professional writers using this system. Further research into this and other literary interfaces could and should explore how they could be taken up in particular educational contexts over longer time-scales of literacy (Lemke 2000). One might reasonably wonder whether training with such tools over periods of time has effects on one’s mode of composition the same way that attending a spin class every week has effects on one’s body. Further research could also focus more closely on the perception of overall “creativity”—whether writers feel as though the system makes them more creative, and whether readers perceive texts written with this system as more creative.

Unexplored too are the political and ideological potentials of this kind of progymnastic exercise. Researchers have begun both to critique and attempt to reverse the biases (especially gender and racial biases) in large data sets and the models trained upon them (Bolukbasi et al. 2016). One could imagine a kind of progymnasmata that targets overly familiar and biased ways of talking about male or female characters, for instance, and encouraging the writer’s departure from stereotypical use of language (such as a tedious insistence that a queen be “fair”; see again Table 1).

Work in computational creativity has focused on how to make creative writing more pleasant, less cognitively and psychologically taxing (Kantosalo et al. 2014; Gonçalves et al. 2017; Gonçalves and Campos 2018). Progym clearly aims to make the task of writing harder rather than easier. Future research could also consider the psychological aspects of users’ interactions with intentionally-critical progymnastic systems and could consider techniques of gamification to motivate writers to engage with them.

Acknowledgements

Thanks to the anonymous reviewers for their thoughtful suggestions.

References

- Bamman, D.; O’Connor, B.; and Smith, N. A. 2013. Learning latent personas of film characters. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 352–361.
- Bird, S.; Klein, E.; and Loper, E. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. ” O’Reilly Media, Inc.”.
- Bolukbasi, T.; Chang, K.-W.; Zou, J. Y.; Saligrama, V.; and Kalai, A. T. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in neural information processing systems*, 4349–4357.
- Dubremetz, M., and Nivre, J. 2015. Rhetorical figure detection: The case of chiasmus. In *Proceedings of the Fourth Workshop on Computational Linguistics for Literature*, 23–31.
- Erasmus, D. 1512–1578. *De Utraque Verborum ac Rerum Copia*. Marquette University Press.

- Gabriel, R. P.; Chen, J.; and Nichols, J. 2015. Inkwell: A creative writer's creative assistant. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition*, 93–102. ACM.
- Gianfortoni, P.; Adamson, D.; and Rosé, C. P. 2011. Modeling of stylistic variation in social media with stretchy patterns. In *Proceedings of the First Workshop on Algorithms and Resources for Modelling of Dialects and Language Varieties*, 49–59. Association for Computational Linguistics.
- Gonçalves, F., and Campos, P. 2018. Enhancing your mental well-being and creativity while writing: A crowdsourced approach. In *IFIP Working Conference on Human Work Interaction Design*, 17–35. Springer.
- Gonçalves, F.; Caraban, A.; Karapanos, E.; and Campos, P. 2017. What shall i write next?: Subliminal and supraliminal priming as triggers for creative writing. In *Proceedings of the European Conference on Cognitive Ergonomics 2017*, 77–84. ACM.
- Hearst, M. A. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, 539–545. Association for Computational Linguistics.
- Hoey, M. 2007. Lexical priming and literary creativity. *Text, discourse and corpora: Theory and analysis* 729.
- Honnibal, M., and Johnson, M. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 1373–1378. Lisbon, Portugal: Association for Computational Linguistics.
- Indurkha, B. 2016. On the role of computers in creativity-support systems. In *Knowledge, Information and Creativity Support Systems: Recent Trends, Advances and Solutions*. Springer. 213–227.
- Kantosalo, A.; Toivanen, J. M.; Xiao, P.; and Toivonen, H. 2014. From isolation to involvement: Adapting machine creativity software to support human-computer co-creation. In *ICCC*, 1–7.
- Kantosalo, A. A.; Toivanen, J. M.; and Toivonen, H. T. T. 2015. Interaction evaluation for human-computer co-creativity. In *Proceedings of the Sixth International Conference on Computational Creativity*. Brigham Young University.
- Kennedy, G. A. 2003. *Progymnasmata: Greek textbooks of Prose Composition and Rhetoric*. Brill.
- Lemke, J. L. 2000. Across the scales of time: Artifacts, activities, and meanings in ecosocial systems. *Mind, Culture, and Activity* 7(4):273–290.
- Lubart, T. 2005. How can computers be partners in the creative process: classification and commentary on the special issue. *International Journal of Human-Computer Studies* 63(4-5):365–369.
- Manjavacas, E.; Karsdorp, F.; Burtenshaw, B.; and Kestemont, M. 2017. Synthetic literature: Writing science fiction in a co-creative process. In *Proceedings of the Workshop on Computational Creativity in Natural Language Generation (CC-NLG 2017)*, 29–37.
- Miller, G. A. 1995. Wordnet: a lexical database for english. *Communications of the ACM* 38(11):39–41.
- Montfort, N., and Fedorova, N. 2012. Small-scale systems and computational creativity. In *International Conference on Computational Creativity*, 82.
- Oliveira, H. G.; Mendes, T.; and Boavida, A. 2017. Copoetryme: a co-creative interface for the composition of poetry. In *Proceedings of the 10th International Conference on Natural Language Generation*, 70–71.
- Oliveira, H. G. 2012. Poetryme: a versatile platform for poetry generation. *Computational Creativity, Concept Invention, and General Intelligence* 1:21.
- Riedl, M. O., and O'Neill, B. 2009. Computer as audience: A strategy for artificial intelligence support of human creativity. In *Proc. CHI Workshop of Computational Creativity Support*.
- Roemmele, M., and Gordon, A. S. 2015. Creative help: a story writing assistant. In *International Conference on Interactive Digital Storytelling*, 81–92. Springer.
- Roemmele, M. 2016. Writing stories with help from recurrent neural networks. In *Thirtieth AAAI Conference on Artificial Intelligence*.
- Smith, A. G.; Zee, C. X.; and Uitdenbogerd, A. L. 2012. In your eyes: Identifying clichés in song lyrics. In *Proceedings of the Australasian Language Technology Association Workshop 2012*, 88–96.
- Swanson, R., and Gordon, A. S. 2008. Say anything: A massively collaborative open domain story writing companion. In *Joint International Conference on Interactive Digital Storytelling*, 32–40. Springer.
- Toivanen, J.; Toivonen, H.; Valitutti, A.; Gross, O.; et al. 2012. Corpus-based generation of content and form in poetry. In *Proceedings of the third international conference on computational creativity*. University College Dublin.
- Veale, T., and Hao, Y. 2007. Comprehending and generating apt metaphors: a web-driven, case-based approach to figurative language. In *AAAI*, volume 2007, 1471–1476.
- Veale, T., and Hao, Y. 2011. Exploiting readymades in linguistic creativity: A system demonstration of the jigsaw bard. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies: Systems demonstrations*, 14–19. Association for Computational Linguistics.
- Veale, T. 2013. Linguistic readymades and creative reuse. *Journal of Integrated Design and Process Science* 17(4):37–51.
- Vygotsky, L. S. 1980. *Mind in society: The development of higher psychological processes*. Harvard university press.
- Webb, R. 2001. The progymnasmata as practice. *Education in Greek and Roman Antiquity* 289–316.

SpaceSheet: Navigating Conceptual Space with a Spreadsheet Interface

Tom White and Bryan Loh

School of Design
Victoria University of Wellington
Wellington, New Zealand

{tom.white@vuw.ac.nz, lohbrya@myvuw.ac.nz}

Abstract

We introduce a new spreadsheet based interface called SpaceSheets for creating novel images and other media. Unlike traditional digital tools, ours is parameterized entirely by a neural network with no pre-programmed rules or knowledge representations. The capability of SpaceSheets to support visual exploration and communication is demonstrated within the context of several domains including facial images, fonts, and english words. SpaceSheets is demonstrated to support the experimentation and exploration of latent spaces enabling more effective design experimentation.

Introduction

Problem solving can be viewed as a search for a solution within a space. In design, this process involves generating solutions and evaluating their consequences relative to goals and constraints (Simon 1995). These experiments are enabled through representations in the form of drawings and diagrams. Computational design tools enable users to construct and manipulate representations digitally. These tools often impose a high cost to design experimentation due to the mismatch between low-level design operations in expressing more abstract design intent.

Generative models learn more compact representations of the training data in a vector space of latent variables. Latent variables are sampled from high-dimensional latent space and can be decoded back into observable values. Additionally, semantic operations can be performed within latent space using vector arithmetic (White 2016).

Spreadsheet interfaces are a ubiquitous part of office productivity suites. They enable users to perform experimental calculations using a set of formulae which define relationships spatially. Automatic recalculation supports experimentation by enabling users to observe the results of their actions immediately and act accordingly.

We developed SpaceSheet (Figure 1) to leverage the familiarity and power of spreadsheet interfaces for the purpose of design experimentation within latent space. It has been adapted to enable non-experts to explore and experiment within latent spaces.

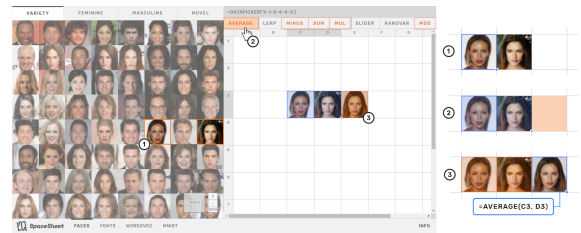


Figure 1: The SpaceSheet being used to perform an average between two latent variables

Background

Conceptual Spaces

Generative models are a popular approach to unsupervised machine learning. Generative neural network models are trained to produce data samples that resemble the training set (Karpathy et al. 2016). Because the number of model parameters is significantly smaller than the training data, the models are forced to discover efficient data representations. These models are sampled from a set of latent variables in a high-dimensional space, called a latent space. Latent space can be sampled to generate observable data values. Learned latent representations often also allow semantic operations with vector space arithmetic (Figure 2), a phenomenon discovered previously in the latent space of language models (Mikolov et al. 2013).

Generative models are often applied to datasets of images. Two popular generative models for image data are the Variational Autoencoder (Kingma and Welling 2013) (VAE) and the Generative Adversarial Network (Goodfellow et al. 2014) (GAN). VAEs use the framework of probabilistic graphical models with an objective of maximizing a lower bound on the likelihood of the data. GANs instead formalize the training process as a competition between a generative network and a separate discriminative network. Though these two frameworks are very different, both construct high-dimensional latent spaces that can be sampled to generate images resembling training set data. Moreover, these latent spaces are generally highly structured and can enable complex operations on the generated images by simple vector space arithmetic in the latent space (Larsen,

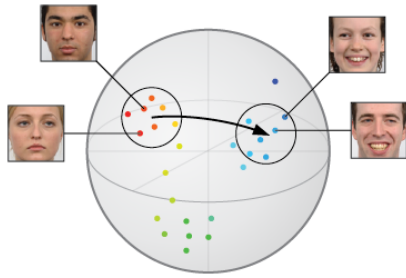


Figure 2: Schematic of the latent space of a generative model. In the general case, a generative model includes an encoder to map from the feature space (here images of faces) into a high-dimensional latent space. Vector space arithmetic can be used in the latent space to perform semantic operations. The model also includes a decoder to map from the latent space back into the feature space, where the semantic operations can be observed. If the latent space transformation is the identity function we refer to the encoding and decoding as a reconstruction of the input through the model.

Sønderby, and Winther 2015).

In the latent space of generative models, many high-level attributes can be represented as a vector (Figure 3). Using techniques from (White 2016), multiple attributes can be decoupled further to create a visualization of possible states across multiple semantic vectors (Figure 4). For example, when trained on a dataset of portraits, latent vectors can be computed for "smiling" and "mouth open" which then applied to new face images.



Figure 3: Traversals along the smile vector using a GAN model from (Dumoulin et al. 2016)

Prior to the discovery of neural network latent spaces supporting semantic operations, cognitive science had hypothesized the existence of knowledge representations that were primarily geometric instead of symbolic. One primary proponent was Gärdenfors who proposed a framework of "Conceptual Spaces" as structured multi-dimensional feature spaces to support modeling information processes such as concept learning and prototype theory (Gärdenfors 2011). Notably, conceptual spaces were proposed as a model of how people structure concepts, independent of any proposed computational implementation of how they might come about.

We adapt the terminology and claim that latent spaces of generative neural networks function as conceptual spaces

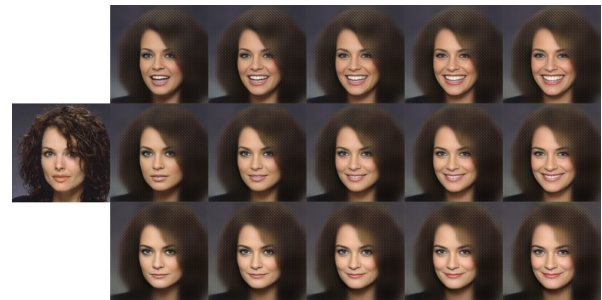


Figure 4: Decoupling attribute vectors for smiling (x-axis) and mouth open (y-axis) allows for more flexible latent space transformations. Input shown at left with reconstruction adjacent. Using a VAE model from (Lamb, Dumoulin, and Courville 2016)

which can be used as non-symbolic knowledge representation layers in other tools. With this framework, we examine the ability of this representation layer built from the latent space of a generative neural network model to support a new type of spreadsheet interface tool. The tool itself is domain independent and is shown to be useful in several domains. In exploring these particular domains, our tool constructs subspaces of the larger conceptual space of possibilities as a parameter space of a spreadsheet driven exploration tool.

Supporting Design Experimentation

Design principles have been identified by (Resnick et al. 2005) and (Terry and Mynatt 2002) for user interfaces to support design experimentation and exploration.

These principles can be summarised by the three user interface requirements proposed in Design Principles for Tools to Support Creative Thinking (Resnick et al. 2005) (paraphrased): It must be very easy to try things out and then backtrack when unsuccessful. Tools should be 'self-revealing' in what they can achieve. Make it very fast to sketch out different alternatives

These principles are supported by (Terry and Mynatt 2002) where they identify three activities in the process of reflection-in-action (Schon 1984) that should be supported by user interfaces for design experimentation. They are: Near-Term Experimentation, Generating Variations, and Evaluation.

Near-Term Experimentation is used to describe actions which intend to "discover and instantiate the next move" (Terry and Mynatt 2002, p. 39). In a user interface, users would make hypotheses about the next action to be made, and test their hypothesis by "invoking a command and adjusting its settings to achieve the imagined effect". The users would then "either accept the command, tweak the parameters more, or undo it and try another tact" (Terry and Mynatt 2002, p. 40).

Variations are created by the designer to explore alternatives deeply. It enables them "to better understand the problem, its boundaries, and potential solutions" (Terry and Mynatt 2002, p. 40). An example of this is where design-

ers make “multiple variations of a specific component by creating them side-by-side on a large canvas ... and iterate on promising versions to arrive at an acceptable solution” (Terry and Mynatt 2002, p. 40).

Users need to evaluate their progress as they work on a task. This happens after near-term experiments, as well as after generating variations: “the moment in which the individual reassesses the problem and their understanding of it, before making the next move” (Terry and Mynatt 2002, p. 40).

Spreadsheet as a Design tool

Spreadsheets may seem like an unlikely design tool. However, the ability to express relationships between cells make it functionally suited to express operations in latent space. Additionally, it satisfies the three user requirements for software to support design experimentation — Near-Term Experimentation, Generating Variations, and Evaluation as proposed by (Terry and Mynatt 2002).

Near-term experimentation is supported by the automatic updating feature of the spreadsheet. Users are able to set up scenarios of logic and calculate the results to ‘what if’ questions instantly by modifying the cell values. This establishes a tight feedback loop between the user’s actions and its implications. When coupled with the ability to undo actions, it enables users to discover and instantiate moves, and backtrack if the results are unsatisfactory.

The generation of variations is supported by enabling users to duplicate instances of data onto other cells within the document. These copies can then be modified independently from the original data.

Evaluation is supported by enabling users flexibility in how they choose to organise data in the document. Users can set up custom templates in a layout which best supports their preferences and the problem to be solved.

In addition to their promise in supporting design experimentation, spreadsheet software is well-established within office productivity suites. Users with an understanding of how conventional spreadsheets function are able to transfer their understanding to the use of the design tool.

SpaceSheet

SpaceSheet consists of a data picker exposing latent variables to operate with and a spreadsheet to define operations between the variables. In both, latent variables are decoded into observable images.

Data Picker

The data picker is a predetermined set of latent variables which have been organized into a grid. The set of variables in the data picker act as the points of reference from which the latent space can be explored from. Diversity has been prioritized in the selected set to maximize the variety of possible outcomes that can be explored. Multiple data pickers have also been implemented as tabs to provide various pre-baked distributions of latent variables.

Spreadsheet

The spreadsheet is the main workspace of the tool. It enables users to express relationships between cells using formulae. Operations between cells containing latent variables are computed with vector arithmetic, and its result is decoded into an image. Common operations can be defined by clicking on buttons at the top of the spreadsheet. These buttons are selection-aware, and highlight to suggest operations based on the selected cells. A live SpaceSheets demo is available online¹ and the appendix contains a list of supported operations and sample workflows.

Applications

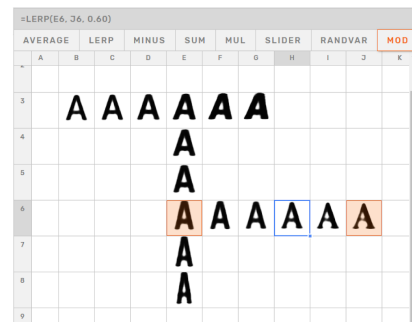


Figure 5: SpaceSheet with Font Model

Initial efforts are focused on experimenting in various domains to encourage the development of a general-purpose model agnostic set of operations. A SpaceSheet to explore a generative model of fonts (Bernhardsson 2015) has been implemented to be used as a design tool (Figure 5). User testing indicated that the tool enabled designers and non-designers alike to explore design variations easily (Loh 2018).

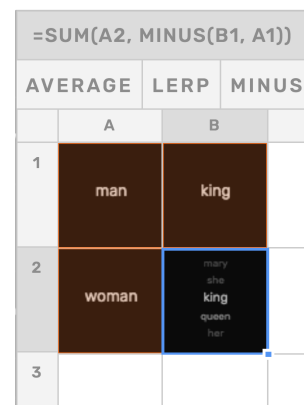


Figure 6: SpaceSheet with word2vec

¹<https://vusd.github.io/spacesheet/>

The concepts have also been extended to domains other than images and with models that are not generative, such as the Word2Vec model (Mikolov et al. 2013). This version of the SpaceSheet can be used to find word analogies and perform interpolations using nearest neighbors (Figure 6).

A SpaceSheet has been created to enable the exploration of the BigGAN model (Brock, Donahue, and Simonyan 2018). In this implementation, the primary DataPicker for this implementation has been curated to enable users to experiment with a variety of image classes (Figure 7).

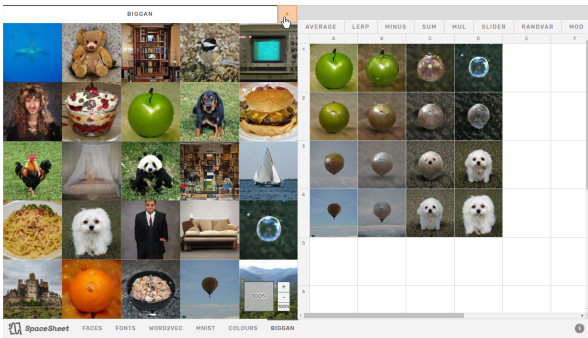


Figure 7: BigGAN SpaceSheet with a generic DataPicker across all image classes

Custom DataPickers of other classes, or combinations of other classes can be created using the DataPicker creator (Figure 8). The DataPicker creator enables users to a) explore and select one or more classes from a searchable, hierarchically organised tree checklist, b) control the amounts of each class to composite in the resulting class, and c) preview example reconstructions of the resulting class before creating a DataPicker of the resulting class. Once created, this new custom DataPicker will be available for use in the spreadsheet (Figure 9).

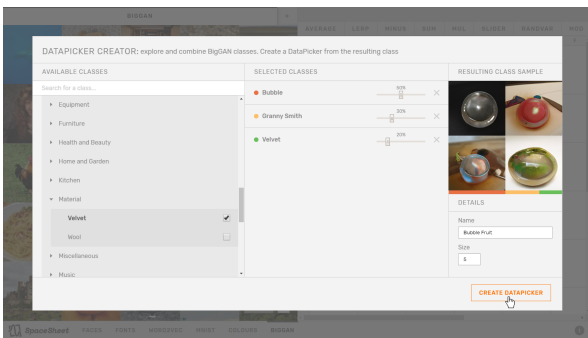


Figure 8: BigGAN SpaceSheet custom DataPicker interface

Evaluation

User testing of SpaceSheets on a model of fonts (Loh 2018) revealed that the tool enabled a novel method to experiment with designs. Users explore design possibilities from a top-down approach by deriving meaning and navigating within a

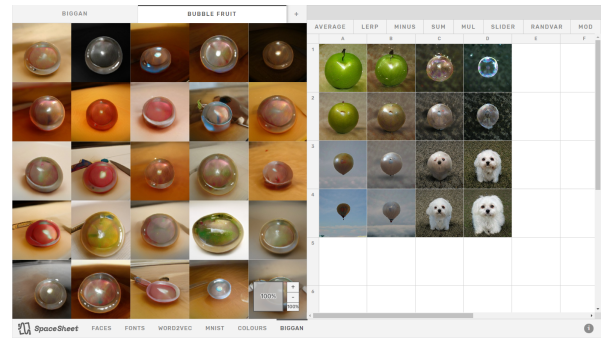


Figure 9: BigGAN SpaceSheet with a custom DataPicker made from combining a user-provided ratio of the "Bubble", "Granny Smith", and "Velvet" image classes

preconstructed model, rather than constructing a model from the bottom-up.

This method of working was reported to be more supportive of design exploration, more efficient, and capable of enabling non-designers to explore design possibilities. Unsurprisingly, it required new skills and intuition to be used to its full effect. A lack of knowledge in deriving and applying attribute vectors from latent space limited users' expressivity and control over their experiments. Due to this, interpolation was found to be the most intuitive and common method to arrive at search targets.

Expressing low-level transformations such as positioning and scale through SpaceSheet often resulted in distorted reconstructions which did not match the expectations of the user. This is attributed to a mismatch in the high-level probabilism of sampling latent spaces is an ill-fit to express concrete design intent. However, this uncertainty has been reported to be serendipitous when distortions in the reconstruction added to the aesthetics of the design.

Discussion

SpaceSheets explores the potential of latent spaces to be used as a tool for design experimentation. The research finds it to enable a novel method to work with designs which supports more efficient, high-level design experimentation to designers and non-designers alike.

User intuition and skill in deriving meaning from latent spaces is fundamental to conduct design experiments with a fine level of control. This intuition can be considered a skill which can be developed through continued experience with the flexible, low-level interface provided by the SpaceSheet. Although latent spaces enable designers to express more meaningful design operations computationally, it provides redundant uncertainty for low-level design operations. It is with this understanding that latent spaces are best considered as a complementary new primitive to build smarter design tools.

References

- Bernhardsson, E. 2015. deep-fonts: Generate fonts using deep learning.
- Brock, A.; Donahue, J.; and Simonyan, K. 2018. Large scale GAN training for high fidelity natural image synthesis. *CoRR* abs/1809.11096.
- Dumoulin, V.; Belghazi, I.; Poole, B.; Mastropietro, O.; Lamb, A.; Arjovsky, M.; and Courville, A. 2016. Adversarially Learned Inference. *ArXiv e-prints*.
- Gärdenfors, P. 2011. Semantics based on conceptual spaces. In *Logic and Its Applications - 4th Indian Conference, ICLA 2011, Delhi, India, January 5-11, 2011. Proceedings*, 1–11.
- Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A. C.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, 2672–2680.
- Karpathy, A.; Abbeel, P.; Brockman, G.; Chen, P.; Cheung, V.; Duan, R.; Goodfellow, I.; Kingma, D.; Ho, J.; Houthoofd, R.; Salimans, T.; Schulman, J.; Sutskever, I.; and Zaremba, W. 2016. Generative models. Technical Report 1, OpenAI.
- Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes. *CoRR* abs/1312.6114.
- Lamb, A.; Dumoulin, V.; and Courville, A. 2016. Discriminative Regularization for Generative Models. *ArXiv e-prints*.
- Larsen, A. B. L.; Sønderby, S. K.; and Winther, O. 2015. Autoencoding beyond pixels using a learned similarity metric. *CoRR* abs/1512.09300.
- Loh, B. 2018. Spacesheets: Design experimentation in latent space.
- Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *CoRR* abs/1301.3781.
- Resnick, M.; Myers, B.; Nakakoji, K.; Shneiderman, B.; Pausch, R.; Selker, T.; and Eisenberg, M. 2005. Design principles for tools to support creative thinking.
- Schon, D. 1984. The reflective practitioner: How professionals think in action.
- Simon, H. 1995. Problem Forming, Problem Finding, and Problem Solving in Design. 245–257.
- Terry, M., and Mynatt, E. 2002. Recognizing creative needs in user interface design. 38—44.
- White, T. 2016. Sampling generative networks. *CoRR* abs/1609.04468.

Appendix: Implementation Details

Supported Operations

Operation	Description	Formula
Sum	Adds a list of numbers / variables	<code>SUM(val1, val2, val3, ...)</code>
Minus	Subtracts two numbers / variables in sequence	<code>MINUS(val1, val2)</code>
Multiply	Multiplies a list of numbers / variables	<code>MUL(val1, val2, val3, ...)</code>
Linear Interpolation	Calculates the value in between two numbers / vectors at a specified amount	<code>LERP(from, to, amount)</code>
Average	Calculates the average of a list of numbers / vectors	<code>AVERAGE(val1, val2, val3, ...)</code>
Distance	Calculates the euclidean distance between two numbers / vectors	<code>DIST(val1, val2)</code>
Modulate	Creates a scrubbing interface which can modulate a cell	<code>MOD(cell, degree, radius)</code>
Random Variable	Creates a random latent variable	<code>RANDVAR(seed)</code>
Slider	Creates a number which is controlled by a slider element	<code>SLIDER(min, max[, step])</code>

Interactive Cell Types

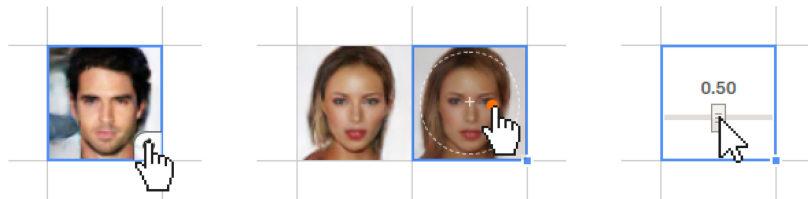


Figure 10: RANDVAR, MOD and SLIDER cells.

Several alternative cell types have been implemented to create interface elements which support more effective search and exploration. These are instantiated by the operations:

`RANDVAR(seed)`

The RANDVAR (random variable) cell instantiates a latent variable from a random seed. This enables users to operate using latent variables beyond the limited set afforded by the Data Picker. A button displays when the cell is hovered over which enables users to randomise the cell directly.

`MOD(base, degree, distance)`

The MOD (modulate) cell exposes a joystick interface which enables users to scrub locally around a given latent variable to arrive at similar latent variables. The degree of difference can be controlled by the joystick's distance from the center of the cell.

`SLIDER(min, max [,step])`

The slider cell enables users to create a number controlled by a slider element.

Example Workflows

Interpolation



Figure 11: An interpolation between two latent variables

Extrapolation

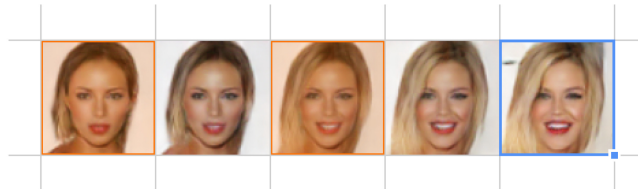


Figure 12: Extrapolating from two points.

Extrapolating from latent variables can be used to emphasise attributes which vary between its anchors. In this example, the difference between the highlighted anchors - blond hair, large smile, etc. - have been emphasised by extrapolating beyond the end anchor.

Averaging

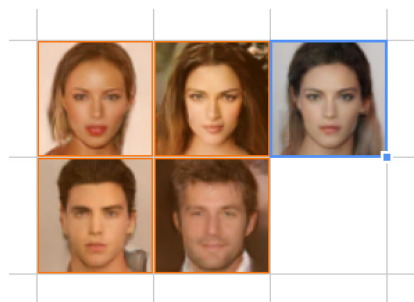


Figure 13: Calculating the average reconstruction of a group of latent variables

Analogy

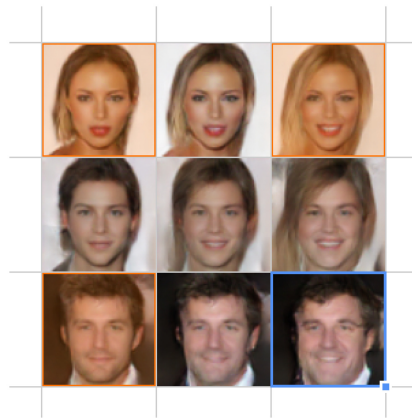


Figure 14: An analogical construction. The bottom right cell applies the difference between the top cells to the cell on the bottom-left

Given three reconstructions (top-left, top-right, bottom-left), the SpaceSheet calculates the bottom-right corner by analogy. This is achieved by applying the difference between the top variables to the bottom-left variable. In this example, a toothy grin has been applied to the man.

Attribute Vectors

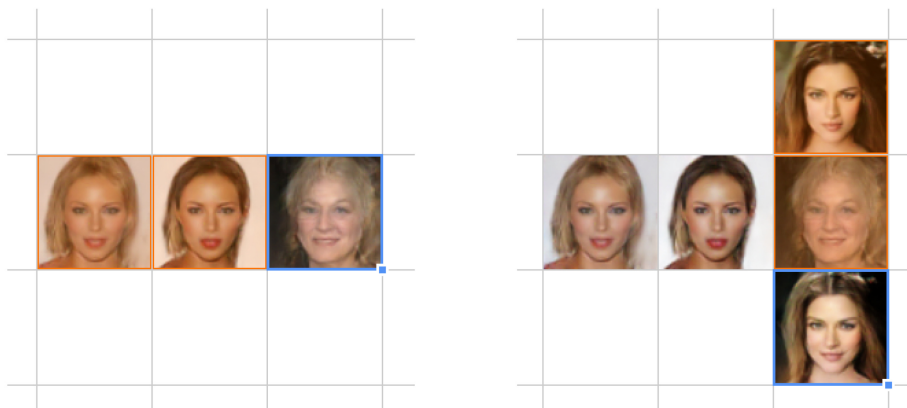


Figure 15: Isolating a 'blonde' vector by subtraction (left). Adding the attribute vector to a new latent variable (right)

Specific attributes can be applied as operations to latent variables. Attribute vectors can be isolated by subtracting a latent vector with desired attributes with one without the attributes. This attribute vector can be added to another latent variable to apply the isolated attribute. The example image shows this two-step process. In the first, a 'blonde' attribute vector has been isolated by computing the difference between the highlighted cells. This vector is then applied in the right image by addition. The result is a more blonde version of the initial latent variable.

Deep Learning in a Computational Model for Conceptual Shifts in a Co-Creative Design System

Pegah Karimi¹, Mary Lou Maher¹, Nicholas Davis¹, Kazjon Grace²

¹UNC Charlotte, ²The University of Sydney

¹USA, ²Australia

pkarimi@uncc.edu, m.maher@uncc.edu, ndavis64@uncc.edu, kazjon.grace@sydney.edu.au

Abstract

This paper presents a computational model for conceptual shifts, based on a novelty metric applied to a vector representation generated through deep learning. This model is integrated into a co-creative design system, which enables a partnership between an AI agent and a human designer interacting through a sketching canvas. The AI agent responds to the human designer's sketch with a new sketch that is a conceptual shift: intentionally varying the visual and conceptual similarity with increasingly more novelty. The paper presents the results of a user study showing that increasing novelty in the AI contribution is associated with higher creative outcomes, whereas low novelty leads to less creative outcomes.

Introduction

Creative systems are computational systems that either model human creativity in some manner or are designed to support and inspire creativity. Over the last few years, three main approaches to these systems have emerged: fully autonomous creative systems, creativity support tools, and co-creative systems. Fully autonomous creative systems, part of the field of computational creativity, are designed to generate creative artifacts or exhibit creative behaviors (Colton et al. 2015; Das and Gambäck 2014). Creativity support tools, on the other hand, are technologies that can support human creativity by accelerating or augmenting some facets of the creative process (Shneiderman 2007; Voigt, Niehaves, and Becker 2012). Finally, co-creative systems incorporate concepts from both fully autonomous systems and creativity support tools: they enable human users and computer systems to work together on a shared creative task (Davis et al. 2015a; Yannakakis, Liapis, and Alexopoulos 2014).

In this paper, we introduce the algorithms for a co-creative sketching tool called the Creative Sketching Partner (CSP), which involves collaboration between a designer and an AI agent on a shared design task. Figure 1 illustrates the CSP tool, in which the design task is described at the top and the three sketches below represent the responses to this task. The two sketches at the top represent the user's initial sketch on the left and the AI agent's responding sketch and label for the sketch on the right. The

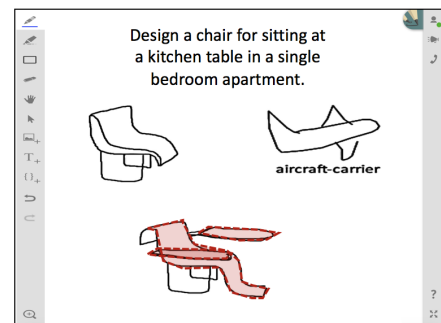


Figure 1: The Creative Sketching Partner interface.

sketch at the bottom of the canvas is the user's new sketch, with the shaded region showing the user's additions inspired by the AI agent's sketch. The system utilizes a computational model of conceptual shifts (Karimi et al. 2018b; 2018a) to guide users toward different aspects of the design space based on the amount of visual and conceptual similarity to the user's sketch input. Visual similarity entails identifying a sketch that shares some structural characteristics, whereas conceptual similarity identifies a concept that has some semantic relationship. We present users with stimuli that have either both high visual and conceptual similarity (like a pen and a pencil) or low visual and conceptual similarity (like a dolphin and a chair).

Karimi et al. (2018c) introduced a framework of ways to evaluate creativity in co-creative systems. It was found that current co-creative systems research tends to focus on measuring the usability of the system, rather than on operationalising creativity. This demonstrates an opportunity for adopting metrics from computational creative systems in order to empower co-creative systems with the capacity to measure the creativity of their contributions to the output. For our conceptual shift model, we adopt one of the most commonly measured components of creativity from computational creative systems: novelty (Grace et al. 2015). Novelty is associated with measuring how different an artifact is compared to another set of artifacts (Grace et al. 2015). The novelty can be based on a comparison with a universal set of artifacts, which we will call a *universal measure*,

or on a set of artifacts that the user has previously experienced, which we will call a *personal measure*. In this paper, we use a universal measure based on a large dataset of labelled sketches and deep learning that enables two kinds of representation: one that enables a measure of visual similarity and one that enables a measure of conceptual similarity. From these metrics, we have constructed a universal composite measurement of novelty that is a combination of the distance between feature vectors in the visual space and the conceptual space.

We hypothesize that, when a system provides stimulus in the form of design concept responses that are highly novel to the user's design, it leads to more transformative creative outcomes. In these cases, the designer is able to draw upon distant visual and semantic features to inspire their creative process, such as adding features from another design domain. In contrast, when the system displays stimulus design concepts that are less novel to the user's design, it corresponds to less creative outcomes. The features of similar designs do not provide highly novel input to the process, leading to design iterations that share many attributes with the designer's original sketch. To explore this hypothesis we performed a user study utilizing a Wizard of Oz system to see how altering the novelty of the AI agent's response affected the creativity of the user's response. Participants experienced three conditions: low, intermediate, and high novelty in the system's response. After the sketching experience, participants were interviewed and surveyed to determine how the AI agent's responses affected their creativity. We found that, based both on our quantitative and qualitative results, the high novelty conceptual shifts stimulated more creative thinking than the low novelty ones.

Related Research

Over the last few decades, digital tools have been introduced as a way to support design creativity (Johnson et al. 2009). These tools offer a variety of functions that allow designers to share their digital sketches and suggest new ideas to facilitate creativity. More recently, intelligent systems have been developed that enable collaboration with designers in real time. These systems, also referred to as computational co-creative systems, work alongside human users to encourage their creativity, support inspiration, and stimulate the user to continue creating. ViewPoints AI (Jacob et al. 2013) is an example of an artistic co-creative system that has applications in dance and theater. It uses a compositional technique that perceives and analyzes human movements and gestures to facilitate an AI response in real time. Morai Maker (Guzdial et al. 2019) is an example of a co-creative game level design tool that assists users in authoring game level content.

Co-creative sketching systems are an active area of research in the computational creativity community. One such example is the Drawing Apprentice, which is a co-creative drawing partner that collaborates with users in real time (Davis et al. 2015b). The system uses sketch recognition to identify objects drawn by the user and selects a complementary object to display on the screen. Complementarity is defined by the semantic distance between the user's sketched object and the target object. DuetDraw (Oh et al. 2018) is

another example of a co-creative sketching tool that works alongside the user by recognizing what the user draws and drawing related content to complete a shared scene. In our work, we use visual and conceptual similarity to select an object from a distinct category to be drawn on the screen in order to support the design process. Instead of selecting a sketch from the same conceptual category, such as Drawing Apprentice, the CSP uses a computational model of conceptual shifts (Karimi et al. 2018b) to determine an appropriate target sketch from a dataset.

Conceptual shifts in design can occur when a sketch of one concept is recognized as being similar to a sketch of another concept (Karimi et al. 2018b). Identifying and capitalizing on conceptual shifts is an important component of the design process, as it allows designers to perceive their design ideas from different perspectives. There are two modes of perception that have been defined in design: *seeing-that* and *seeing-as* (Suwa and Tversky 1997). Seeing-that refers to the concrete properties of a sketch and their function in the overall design, whereas seeing-as refers to interpretation, in which sketch elements can be considered through multiple perspectives. Conceptual shifts have the potential to inspire designers to adopt the seeing-as mode of perception, exploring how their emerging design could be connected to a variety of distinct concepts presented as stimuli.

Identifying conceptual shifts could also help users overcome design fixation (Purcell and Gero 1996). Designers often have a hard time disengaging from the ideas they developed and learned over time. This effect, called fixation, may be reduced by presenting designers with a sketch of another object that shares some visual and conceptual information. We presume that, when presenting a conceptual shift successfully triggers seeing-as perception, a designer could be distracted from fixation, and potentially develop novel contributions to their design. This could lead to the discovery of innovative solutions for a design task.

The study of creative design has led to a characterization of different types of creativity. Gero (2000) has introduced six forms of design creativity that can form the basis for computational aids: combination, exploration, transformation, analogy, emergence, and first principles. Combination happens when two distinct design concepts are added. Exploration relates to changing some variable values associated with a design concept. Transformation involves altering one or more variables of a design concept through external processes. Analogy is characterized by mapping between structural elements of two dissimilar objects. Emergence occurs when extensional properties of a design concept are identified beyond the intentional ones. First principles use computational knowledge to relate function to behaviour and behaviour to structure. The CSP introduced in this paper can be considered a computational aid to design that can support the first four of these forms of creativity in a co-creative design context: combination, exploration, transformation, and analogy.

Quantifying Conceptual Shifts

Quantifying conceptual shifts is challenging because concepts are not typically represented or evaluated numerically.

Our premise is that the larger the shift, the more creative the resulting design. In order to quantify the scale of a conceptual shift between two sketches (in our case the user's sketch and the system's proposed response), we need a representation space in which we can measure similarity or novelty. The more similar the second sketch is to the first, the less novel the second item is and (we hypothesize) the less likely that it will trigger a conceptual shift. When the two items are less similar, the more novel the stimulus and (again, we hypothesize) the more likely it will result in a conceptual shift.

We focus on novelty in generating conceptual shifts because it has been shown to be a key component in predicting creativity (Grace et al. 2015). The assumption in measuring novelty is the existence of a representation that allows objective measurement of difference. In (Grace et al. 2015), the corpus of designs in the design space were represented as a set of features that formed the basis for correlation and regression analysis. The feature set was extracted from a database in which the information about the designs was manually entered as a set of features with categorical and numerical values. This representation enabled various ways to measure novelty, but not a single novelty score.

In the CSP, we measure novelty by comparing two sketches: an initial sketch presented by the user and a second sketch selected from a large dataset of sketches. Novelty is a combination of two components: the visual similarity based on the visual data and the conceptual similarity based on the label for the sketch. We use deep learning models to extract a vector representation in two design spaces: a visual space using a large dataset of sketches, and a semantic space using a word embedding model. We consider the novelty to be a combination of the classification of visual novelty in the visual space and conceptual novelty in the word embedding space.

We classify novelty into three categories: low, intermediate, and high. Low novelty occurs when two sketches share a large amount of visual and conceptual information, intermediate novelty is when two sketches share some visual and conceptual information, and high novelty occurs when two sketches share little visual and conceptual information. We presume that low novelty lies within the expectation of the user, and that the system's response might be most likely to help the designer add more details to their initial design. Intermediate novelty could instead inspire the designer to explore possible new design ideas associated with their initial design. High novelty has the potential to widen the user's thinking process, making it more likely to help them incorporate new design features from a completely different design space. Based on this presumption, we hypothesize that increasing the novelty of the CSP stimulus will correlate with more creative outputs.

Conceptual Shift Algorithm

In this section we describe an AI model of conceptual shifts. The model selects an object from a database of sketches to be displayed on the canvas as a stimulus during a co-creative session. Our model has two components: visual similarity and conceptual similarity. Visual similarity recognizes pairs

of sketches from distinct categories that share some underlying visual information. Conceptual similarity identifies the semantic similarity between the labels of the sketches.

Figure 2 shows the computational model the AI agent uses to select a sketch of the desired level of novelty in response to the user's input. The visual similarity module computes the distances between the cluster centroids of distinct categories and maps the user's input to the most similar sketches from categories to which it does not belong. The conceptual similarity module takes the pairs of selected category names from the previous step and computes their semantic similarity. In this section, we describe how CSP generates a numerical value for visual and conceptual similarity and determines the conceptual shift candidates based on high, intermediate, and low novelty.

Visual Similarity Module

The visual similarity module uses a large public dataset of human-drawn sketches, called QuickDraw! (QD) (Jongejan et al. 2016), with more than 50 million labeled sketches grouped into 345 categories. In preparation for calculating visual similarity, we have 2 steps: a learning step and a clustering step. In the learning step, the sketches are used to build a vector representation of the sketch's features. In the clustering step, we use the resulting feature vectors for sketches in each category to create clusters of visually similar sketches. This process provides a feature vector representation for calculating the novelty between the user's initial sketch and sketches in the QD dataset using visual similarity.

Deep Learning Model of Sketches for Visual Similarity

As in the case of natural images, sketches can also be processed as a grid of pixels, (h, w, d) , in which h is the height, w is the width, and d is the number of channels. However, in this case, d will be 1 because the sketches are monochrome. To develop a representation for visual similarity we employed a convolutional neural network (CNN) model due to their success in providing high level visual information and discriminating visual appearances, such as shapes and orientations (LeCun, Bengio, and Hinton 2015). We started with a pre-trained model, VGG16 (Simonyan and Zisserman 2014), with 13 convolutional layers, two fully connected layers, and a softmax output layer. The model is primarily trained on the ImageNet dataset (Deng et al. 2009) that contains more than 20 million labeled natural images. We then fine-tune this model on the QD dataset with the objective of classifying a sketch into one of the 345 categories. We use 30,000 training samples and 10,000 validation samples per category, and trained for 1.5 million training steps. Observation shows that the accuracy reaches 52.1% after 1 million steps and remains the same afterwards. We extract a neural representation of each sketch by taking the output of the first fully connected layer, for 4096 values per sketch. However, this model has low accuracy and a high computational cost because of the large number of parameters in the VGG16 architecture and processing sketches as a grid of pixels.

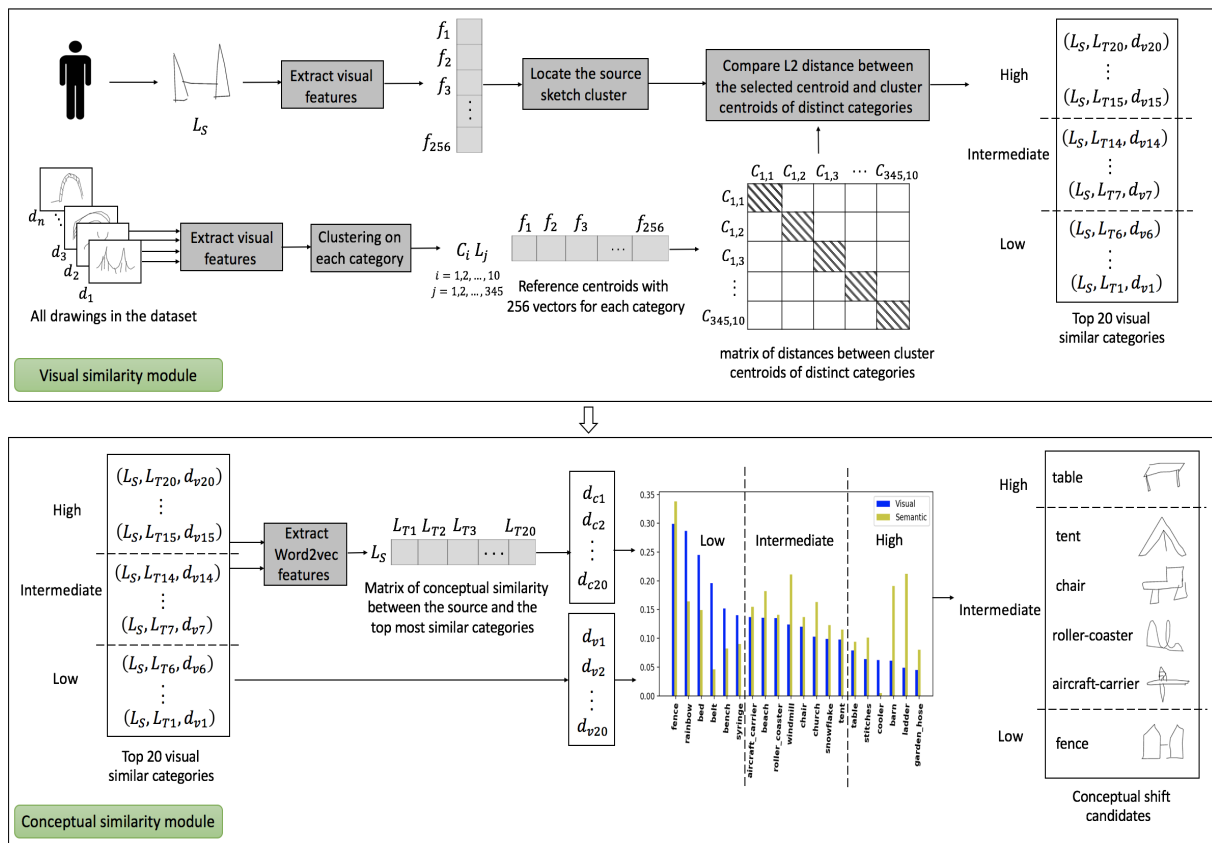


Figure 2: Computational steps for identifying conceptual shifts. Top: Identifying visually similar categories to the user’s input. Bottom: Balancing visual similarity with conceptual similarity and identifying conceptual shifts with high, intermediate, and low novelty.

In order to solve this problem, we tried another representation of sketches: a sequence of pen strokes, inspired by the work done by Ha and Eck on Recurrent Neural Network drawing (Ha and Eck 2017). In this case, each stroke is a list of points with 3 elements: $(\Delta x, \Delta y, p)$. Δx and Δy are the coordinates with respect to the previous point, and p is a binary number that determines whether the stroke is drawn or not (i.e. just moves the pen). Here we use a deep learning model called Convolutional Neural Network-Long Short Term Memory (CNN-LSTM) (Carbone 2017). The model has three one-dimensional convolutional layers and three LSTM layers. We train the model from scratch on the QD dataset with the same objective, training, and validation samples as the CNN-only model. Results show that, after 1 million training steps, accuracy reaches 73.4% and remains the same afterwards. Each sketch is represented by the last LSTM layer, for 256 values per sketch. Table 1 summarizes the results for accuracy and the average-per category inference time for both models. Accuracy measures a true positive rate, while inference time represents the total amount of time it takes to extract features from all sketches of a category. The CNN-LSTM model is clearly both faster and more accurate, and we use it hereafter.

Clustering visually similar sketches in each category

The sketches in a category exhibit a large variability visually. For our visual similarity measure to be meaningful, we group the sketches in each category into clusters and use the feature vector of the cluster centroid as the representative sketch. This process is a form of denoising, where the intra-cluster variability is suppressed. We perform clustering using a K-means algorithm and determine the optimal number of clusters via the elbow method. By analyzing the variance versus the number of clusters, we observed that for most categories the optimal number of clusters is between 7 and 12—we set the number of clusters to 10 across all categories. The distances between the cluster centroids from distinct categories are computed and stored in a matrix of size 3450×3450 : 10 clusters of sketches for each of 345 categories.

Given the source sketch and label from the user, L_S , we first extract visual features using the pre-trained CNN-LSTM model that produces 256 values. We then locate the representative cluster within its category (according to the label of the user’s sketch) by selecting the closest centroid based on the L2 (i.e. Euclidean) distance. Using the distance

	VGG-16	CNN-LSTM
Accuracy	52.1%	73.4%
Inference time	18,000S	960S

Table 1: Classification accuracy and the inference time using two different deep learning models.

matrix, we then select the top 20 most visually similar target clusters from other categories, L_T , as the ones with minimum distance from the representative cluster. The similarity is computed as $1 - d_v$, where d_v is the Euclidean distance normalized across the most visually similar candidates. As the similarity values for the selected target sketches change smoothly, we classify those that fall in the top 33rd percentile of the distribution as low novelty, between 33rd and 66th percentile as intermediate novelty, and above 66th percentile as high novelty.

Conceptual Similarity Module

The conceptual similarity module uses a word embedding model (Mikolov 2016) trained on the Google News corpus with 3 million distinct words. The visual similarity module provides a set of candidate sketches to the conceptual similarity module based on the categories of low, intermediate, and high novelty. We extract the word2vec word embedding features (Mikolov 2016) from these category names. The similarity between the category of the source sketch and the selected target sketch is computed as $1 - d_c$, where d_c is the cosine distance between the feature vectors of category names. The larger number indicates that the two sketch categories are more likely to appear in the same context, whereas a smaller number indicates that the two are less associated with each other. In order to determine the conceptual shift categories, we select those where the visual and conceptual similarity are both high, medium, or low. This is done by selecting candidates for which the difference between visual and conceptual similarity values are below 0.05 and the overall similarity component is computed as the average of visual and conceptual values.

User Study

We conducted a user study to evaluate the effectiveness of our conceptual shift model in a co-creative design session. We investigated how the novelty of the system’s response could inspire user creativity and correspond to different types of design behaviors. Our hypothesis is that increasing the novelty of the system’s response can help designers add new features and/or functions from another design space to their initial drawing, thus leading to more creative outcomes. By contrast, when the system is in the low novelty condition, the designer is presented with the similar features to the initial drawing, which leads to less creative outcomes.

In this study, we used a within-subjects design, such that each participant experienced three conditions with a two-minute break between them. In the first condition the design task is a chair, and the system produces a result that

is highly novel with respect to the participant’s sketch. In the second condition the design task is a streetlight, and the system produces a result associated with intermediate novelty. In the third condition the design task is a bridge, and the system produces a result that is classified as low novelty. Participants were not aware whether they were in a high, intermediate, or low novelty condition. A context is provided to help guide each design task, such as “draw a streetlight for safety at night on a city street of a small town.” When the system’s output object is presented to the user, it is accompanied by a label indicating what the object is. Each design task takes approximately 7 minutes. The order of the three conditions for each participant was randomized to account for any ordering effects.

We used an online sketching tool, called SketchTogether (Bonazza 2019), that enables multiple users to contribute to a shared canvas in real time. This application allowed us to run a Wizard-of-Oz interaction for the user study in which we used the results of the deep learning model for determining high, intermediate, and low novelty sketches, but a person performed the interaction of placing the selected sketch on the shared canvas. Participants underwent a 5-minute training session that included an explanation about the interface tool and the design tasks. After training, participants are asked to start the first design task. The instruction given to the participants were to draw an object according to the design task and iterate on that drawing based on inspiration from the system’s response to their sketch. Following each experimental condition, we asked participants Likert scale survey questions associated with that design session. The questions we asked after each task were:

1. Did the system’s sketch response inspire you to come up with creative ideas for your design objects?
2. Did the system’s sketch response lead you to come up with a different type of design object?

The answers to the survey questions were recorded for quantitative analysis. After the last design session, we asked participants the following questions in an interview:

1. How did the sketches presented by the system affect your creative process?
2. Was it more helpful when the sketches presented by the system were more or less similar to your input?
3. In which of the three design tasks did the system’s sketch inspire you most?
4. Do you have any comments for participating in this study?

The answers to the interview questions were used for qualitative analysis. The entire session for each participant took almost 30 minutes.

Results

The user study included 24 participants recruited from the College of Architecture at a public university in North America. Gender distribution was 15 males and 9 females. The criterion for participating was whether students perform sketching frequently for their design practice. We recorded survey and interview responses for all participants. In this

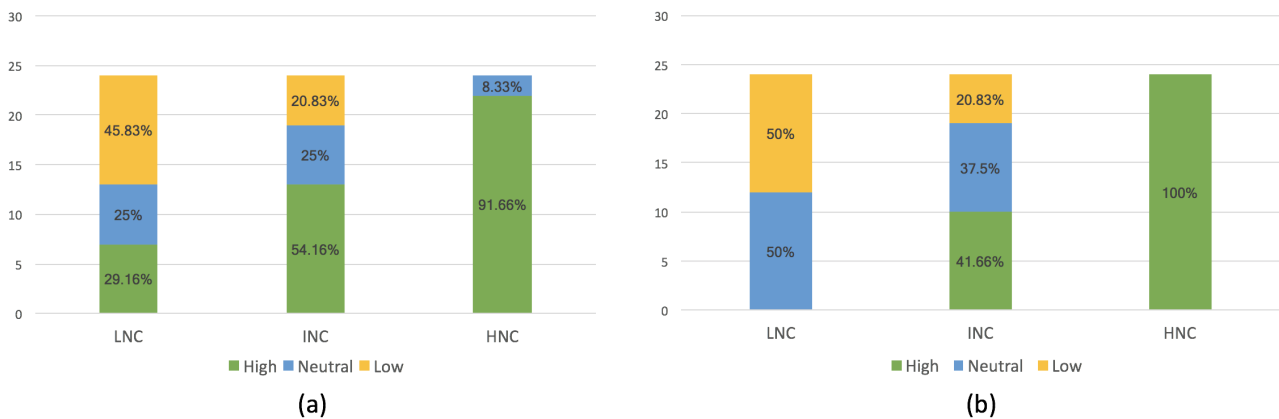


Figure 3: The total percentage of high, intermediate, and low survey responses for (a) inspired creative ideas, and (b) led to different design.

section, we describe our analysis based on the participants' responses in order to investigate our hypothesis.

Quantitative Analysis

We compared the results from the user's feedback on the three design tasks associated with high, intermediate, and low novelty conditions. We grouped the responses into high, neutral, and low ratings: 4 and 5 are considered high, 3 is neutral, and 1 and 2 are low. For each condition we count the number of ratings based on this grouping.

Analysis of creative ideas

Participants were asked to rate the responses provided by the system after each design session. With this question, we aimed to understand whether increasing the novelty of the system's response inspired their creative thoughts. We found that 91.66% of the participants thought that the system's response inspired creativity when the system was in the high novelty condition (HNC) compared to 29.16% in the low novelty condition (LNC). These results indicate that when the system's response is more novel with respect to the user's sketch (HNC), it is associated with more creative outcomes, which may encourage the user to come up with new design ideas for their initial drawing. When the system was in intermediate novelty condition (INC), 54.16% of the participants were highly inspired by the system's response. Figure 3a shows the distribution of the ratings for the three conditions.

Analysis of design object inspiration

Transformational creativity happens when a designer changes one or more structural variables of the current design object to produce new variables (Gero 2000). This implies that the system's response has the potential to inspire the user to transform some features of a design concept by adding new features from another design space related to the system's response. We explored whether increasing the novelty of the system's response can lead to transformational

creativity in which the participant's designed object significantly deviates from their initial sketch. All participants rated high in response to changing their design when the system was in HNC. This indicates that when the system's response was less similar to the participant's input (HNC), they were able to transform their initial sketch. By contrast, when the system was in LNC, none of the participants reported that the system helped them come up with a different type of design object. When the system was in INC, 41.66% of the participants rated high in response to changing their design and 58.33% rated low or neutral (see Figure 3b).

Qualitative Analysis

To understand how the novelty of the system's response can help designers come up with creative ideas for their initial task we analyzed the participants' responses to the interview questions conducted after the design tasks were complete. We aimed to explore the relationship between stimulus novelty and design thinking.

Thematic Analysis

We performed a thematic analysis of the responses the participants gave to the interview questions. Overall, three main themes were found from the interview answers.

- The tool helps with the design process
- High novelty helps changing the design
- Low novelty helps completing the design

In the following section, we elaborate on each of these themes.

Supporting the design process

Most participants found the tool useful, as it can help with the design-thinking process as well as iterating and generating new design ideas. P11 exemplifies how the sketching tool helped their design process, "The sketches presented after I did my initial sketch, change the creative process, making me think of different object and using that design philosophy and then the second object to affect the first." This

participant described how the system's output sketch helped them think of different design ideas and iterate on their initial design sketch. This demonstrates that the tool generally supports the iterative nature of the early design process. Additionally, P14 comments: *"it sort of help[ed] me to see how I think about design, like they teach us just to design, I never really thought about how I go about that process of designing and so having this sort of precedent to work with is more useful to me."* This participant shows the role such a tool could play in design education. It helps to provide precedents that can inform the design process and inspire additional thinking on the topic.

P4 described how helpful the system is when they say, *"I think the system's response is very helpful, because it gives me a leverage on adding to my initial design or just give me some clue or hint to change my design to make it better."* Here, the participant comments about how the tool helps them iterate on their design by adding or changing different elements of the initial sketch based on the 'clues' or 'hints' provided by the system's output. P5 agrees with this sentiment when they said, *"the way that we communicate is great because you add something and I am going to redesign it and so it's great."* This participant focused on the communication channel established between the user and system, and described how this channel helped in the redesign process. In a similar vein, P25 describes how *"it kind of guided me through some conventional ways of improving my design"* which shows how the tool serves to shepherd users through the design process by providing new avenues to explore and inspiration to change the user's initial design.

High novelty inspires changing the design

We found that high novelty conceptual shifts inspire participants to change the overall shape of their design by adding new features from another design space related to the target sketch. In this condition, 21/24 reported that it is more inspiring when the system's response is less similar to their initial design. P11 commented: *"I think to create an interesting result it was more helpful to have a dissimilar object as opposed to a similar, because it allows you to change the form and different ideas instead of just kind of a similar shape affecting it."* This participant indicates that when the system's response is less similar to their initial design (high novelty condition), it helps to change the structure, such that it is possible to incorporate different ideas from the target sketch. Similarly, P10 commented: *"It was easier to make changes when it was more different. I think when something is already similar sometimes my brain already has a same set of ideas, but when I am presented with something different the contrast helps me to generate a new idea."* This participant was able to come with a new idea when he/she was presented with a sketch that was less similar to the initial drawing.

When P16 was presented with a sketch of an aircraft-carrier after designing a chair, they described how the system's sketch opened up new possibilities for them, *"The aircraft-carrier may have chairs but it doesn't elicit specific form especially giving the prompt that is going to be at the kitchen table. Thinking about new possibilities that can hap-*

pen definitely opens the new design criteria." This example shows that the chairs of the aircraft-carrier introduced new design criteria that inspired the participant to sketch a new kitchen chair with the features of aircraft-carrier seats, such as more comfort. Additionally, when P21 was presented with a sketch of a speedboat after designing a chair, they also found new possibilities in the design space, *"The relationship between the two, even though they are used both in the same task or same function because of the difference that one is on water, one needs to be outdoor, the different needs and purposes between the two was influencing me better to create something new between them."* Similarly, P22 used the features of the system's response to reason about their initial sketch, *"The aircraft, because of its curves and the materiality, so thinking about the skin of the material, maybe thinking about its curves so that led me to think about the curves which maybe helped me to think of armrest."* In this example both the structure and the concept of the target sketch inspired the participant to change the shape to be curvy as well as adding new functionalities such as armrests.

Low novelty helps complete the design

Overall, 3/24 participants commented that it is more helpful when the sketch that is presented to them is more similar to their initial drawing (low novelty condition). P4 explains why the sketch of fence that was highly similar to their initial drawing of bridge was more helpful, *"because there were clear features and structures that could help by adding, mainly the similar features."* In this case, the participant preferred to finalize the original drawing by adding more details and structures rather than changing the existing features. Similarly, P9 commented: *"I like the product of end results when stuff [is] more similar. Because I could pull from the profile of fence and add to the bridge...So, you take something from it and add it to your design."* From both P4 and P9, we can conclude that when the system is in low novelty mode the designer mainly adds more details to the initial drawing rather than transforming the shape or adding new features to the drawing. Most participants found the low novelty condition less helpful. For instance, P12 described how they liked less similar designs, *"I would say it was more helpful when it was less similar because then you are not just copying the instances from the other design."* P8 agreed with this sentiment when they said: *"high similarity is kind of within my expectation."*

In both cases of P8 and P12, the low novelty conceptual shift designs do not help to significantly change the original drawing. Instead, they are used to combine some elements of the two sketches. P13 echoes this general viewpoint when they said: *"I think if you are presenting something that is almost exactly the same, you are going to introduce the same idea again."* Similar to P8, this participant also emphasizes that low novelty conceptual shifts are within their expectation. P22 also commented: *"I feel that similar designs didn't give me as much creative freedom."* These examples demonstrate that low novelty conceptual shifts may help to combine the elements of the two sketches, rather than encouraging the user's creative thoughts. Both likely have a role in co-creative design systems, serving different purposes.

Conclusion

This paper presents a computational model of conceptual shifts for a co-creative design system called the Creative Sketching Partner. The tool is meant to inspire design creativity by presenting a sketch of a distinct category that shares some visual and conceptual information with the user's input sketch. We describe the role of deep learning in creating a representation space for measuring distance between the visual and conceptual features of a sketch. We have detailed the process for classifying potential response sketches as low, intermediate, or high novelty with respect to the designer's sketch. A user study is presented in which the participants are given a design task and then experience three different versions of the tool: low, intermediate, and high novelty responses. Both quantitative and qualitative results from the user study demonstrate that the high novelty conceptual shift designs inspire creative thinking more than the low novelty condition.

Acknowledgements

The research reported in this article is funded by NSF IIS1618810 CompCog: RI: Small: Pique: A cognitive model of curiosity for personalizing sequences of learning resources.

References

- Bonazza, M. 2019. SketchTogether. <https://sketchtogether.com>. Accessed: 2019-02-27.
- Carbune, V. 2017. Recurrent neural networks for drawing classification. https://www.tensorflow.org/tutorials/sequences/recurrent_quickdraw.
- Colton, S.; Halskov, J.; Ventura, D.; Gouldstone, I.; Cook, M.; and Ferrer, B. P. 2015. The painting fool sees! new projects with the automated painter. In *ICCC*, 189–196.
- Das, A., and Gambäck, B. 2014. Poetic machine: Computational creativity for automatic poetry generation in bengali. In *ICCC*, 230–238.
- Davis, N.; Hsiao, C.-P.; Popova, Y.; and Magerko, B. 2015a. An enactive model of creativity for computational collaboration and co-creation. In *Creativity in the Digital Age*. Springer. 109–133.
- Davis, N. M.; Hsiao, C.-P.; Singh, K. Y.; Li, L.; Moningi, S.; and Magerko, B. 2015b. Drawing apprentice: An enactive co-creative agent for artistic collaboration. In *Creativity & Cognition*, 185–186.
- Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database.
- Gero, J. S. 2000. Computational models of innovative and creative design processes. *Technological forecasting and social change* 64(2-3):183–196.
- Grace, K.; Maher, M. L.; Fisher, D.; and Brady, K. 2015. Modeling expectation for evaluating surprise in design creativity. In *Design Computing and Cognition'14*. Springer. 189–206.
- Guzdial, M.; Liao, N.; Chen, J.; Chen, S.-Y.; Shah, S.; Shah, V.; Reno, J.; Smith, G.; and Riedl, M. 2019. Friend, collaborator, student, manager: How design of an ai-driven game level editor affects creators. *arXiv preprint arXiv:1901.06417*.
- Ha, D., and Eck, D. 2017. A neural representation of sketch drawings. *arXiv preprint arXiv:1704.03477*.
- Jacob, M.; Coisne, G.; Gupta, A.; Sysoev, I.; Verma, G. G.; and Magerko, B. 2013. Viewpoints ai. In *Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*.
- Johnson, G.; Gross, M. D.; Hong, J.; Do, E. Y.-L.; et al. 2009. Computational support for sketching in design: a review. *Foundations and Trends® in Human-Computer Interaction* 2(1):1–93.
- Jongejan, J.; Rowley, H.; Kawashima, T.; Kim, J.; and Fox-Gieg, N. 2016. The quick, draw!-ai experiment. *Mount View, CA, accessed Feb 17:2018*.
- Karimi, P.; Davis, N.; Grace, K.; and Maher, M. L. 2018a. Deep learning for identifying potential conceptual shifts for co-creative drawing. *arXiv preprint arXiv:1801.00723*.
- Karimi, P.; Grace, K.; Davis, N.; and Maher, M. L. 2018b. Creative sketching apprentice: Supporting conceptual shifts in sketch ideation. In *International Conference on Design Computing and Cognition*, 721–738. Springer.
- Karimi, P.; Grace, K.; Maher, M. L.; and Davis, N. 2018c. Evaluating creativity in computational co-creative systems. In *ICCC*, 104–111.
- LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *nature* 521(7553):436.
- Mikolov, T. 2016. word2vec: Tool for computing continuous distributed representations of words. <https://code.google.com/p/word2vec>.
- Oh, C.; Song, J.; Choi, J.; Kim, S.; Lee, S.; and Suh, B. 2018. I lead, you help but only with enough details: Understanding user experience of co-creation with artificial intelligence. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 649. ACM.
- Purcell, A. T., and Gero, J. S. 1996. Design and other types of fixation. *Design studies* 17(4):363–383.
- Shneiderman, B. 2007. Creativity support tools: Accelerating discovery and innovation. *Communications of the ACM* 50(12):20–32.
- Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Suwa, M., and Tversky, B. 1997. What do architects and students perceive in their design sketches? a protocol analysis. *Design studies* 18(4):385–403.
- Voigt, M.; Niehaves, B.; and Becker, J. 2012. Towards a unified design theory for creativity support systems. In *International Conference on Design Science Research in Information Systems*, 152–173. Springer.
- Yannakakis, G. N.; Liapis, A.; and Alexopoulos, C. 2014. Mixed-initiative co-creativity. In *FDG*.

Read Me Like A Book:

Lessons in Affective, Topical and Personalized Computational Creativity

Tony Veale

School of Computer Science, University College Dublin, Dublin, Ireland.

Abstract

Context is crucial to creativity, as shown by the significance we attach to the labels *P*- and *H*-Creativity. It is context that allows a system to truly assess novelty, or to ensure that its topical artifacts really are topical. An important but an often overlooked aspect of context is personality. A CC system that is designed to reflect a specific aspect of the creative temperament, whether it is humour, arrogance or whimsy, must stay true to this assumed personality in its actions. Likewise, a system that creates artifacts that are rooted in emotion must be sensitive to the personality or mood of its audience. But here we must tread carefully, as the assessment of personal qualities often implies judgement, and so few of us like to be judged, especially by our machines. To better understand the upsides and pitfalls of topicality- and personality-based CC systems, we unpack three of these systems here, and explore the lessons they offer.

The Wonder of You

Creativity can be an intensely personal affair. We put ourselves into what we create, relying on our experiences and values to build artifacts we hope others will value too. In doing so, we reveal our personalities. When we create for others and assimilate the values of an audience, creativity becomes personal and personalized. While it is a tenet of *computational creativity* (CC) that an agent need not be a person to be creative, a creative system may nonetheless need a personality, or an appreciation of the personalities of others, before it can create like a human (Colton *et al.* 2008). ‘*Software is not human*,’ to quote the CC refrain, yet CC systems must appreciate what matters to a human. In any case, we can only know a CC system’s personality, and it can only know ours, by what we do or say, making personal/personalized CC a special case of contextual CC. For CC systems that create in language, context is itself a linguistic artifact, as rooted e.g., in our social media timelines. Here we describe how best to use linguistic context to deliver various forms of topical and personalized CC.

Specifically, we will explore the role of linguistic context in the operation of three *Twitter*-based CC systems, ranging from one that uses context to ensure topicality to ones that view context as the imprint of a user personality.

For personalization, the *Twitter* footprint of a target user – whether their official bio or their recent tweets – offers a textual context in which to situate the generation process. For topical creativity, the aggregated timelines of an array of online news sources, from a *Twitter*-addicted president to the breaking headlines of mainstream media, provide a dynamic context for machine creativity. We explore three modes of CC via these systems: a marriage of linguistic and artistic creativity that maps the digital personality of a user, as reflected in what they tweet, into metaphors that are both textual and visual; a topical creator that generates metaphors for news stories rather than news readers; and a book recommender system that leavens its user-tailored suggestions with humour, and which invents its own book ideas to supplement the titles in its well-stocked database.

The principle that unites all three systems is the role of information compression in CC. One space of information may be compressed or decompressed to yield others, and produce insightful generalizations or vivid elaborations in the process. Thus, compression is required to map a news story to a linguistic metaphor, as the metaphor need only capture the gist of the story. In fact, such information loss is desirable when it leads to generalization and ambiguity, as metaphors should be objects of profound wonderment. When moving from online user personalities to metaphors we require the opposite, *decompression*, to inflate a low-dimensional space of personality types into an elaborate, high-dimensional space of possible character metaphors. Current sentiment analysis techniques can place a user in a space of a dozen or so psychological dimensions, while metaphors will occupy a space that – even after a process of dimension reduction – has hundreds of dimensions. In fact, even the extraction of psychological dimensions is a compression process, since the textual timelines that feed into sentiment analysis are converted into high-dimension distributed spaces built with word co-occurrence statistics.

In the next section we focus on personality-driven CC with a system that maps recent user moods into metaphors and pictures. Our approach is data- and knowledge-based, marrying textual data from a user profile with a symbolic model of the cultural allusions that underpin a machine’s metaphors. Following that, we give statistical form to the notion that metaphors reside in a space of possibilities, so

as to re-imagine metaphor creation as a mapping from one space, a topic model of the news, into another, a space of metaphors that shares exactly the same dimensions. These are whimsical systems that make sport of news and mood, so we present one more system, a CC book recommender that uses simple information-retrieval techniques to guide its suggestions, but which also uses machine creativity in some unobvious ways. This specific system was built for a recent science communications event, and user feedback offers us some lessons on the willingness of humans to be judged by machines. Although whimsy can diminish the severity of a perceived criticism, humour must be wielded with care by our autonomous CC systems, especially if it is unbidden, or used for the furtherance of serious goals.

Metaphor Mirror On The Wall

Consider the problem of generating apt metaphors for the news. As a story breaks and headlines stream on Twitter, we want our metaphor machine to pair an original and insightful metaphor to each headline. So a headline about extreme weather might be paired with a metaphor about nature's destructive might, or a political scandal might be paired to a crime metaphor. As metaphor theorists often speak of multiple spaces – e.g. Koestler (1964), Lakoff & Johnson (1980) and Fauconnier & Turner (2002) all see different viewpoints as different spaces – it is tempting to model each space in a metaphor with its own vector space model (VSM), by equating vector spaces with conceptual spaces. Yet this analogy is misleading, as different VSMs – constructed from different text corpora – must have different dimensions (even if they share the same *number* of dimensions) and we cannot directly perform geometric comparisons between the vectors of two different VSMs. Since the principal reason for building a VSM is the ease with which semantic tests can be replaced with geometric ones, we should build a single vector space that imposes the same dimensions on each conceptual metaphor space. It is useful then to view news headlines and metaphors as comprising two overlapping subspaces of the same VSM.

For a news subspace we collect a large corpus of news content from the Twitter feeds of CNN, Fox News, AP, Reuters, BBC and New York Times, and use a standard compression technique – such as LDA (*Latent Dirichlet Allocation*; Blei *et al.*, 2003), LSA (*Latent Semantic Analysis*; Landauer & Dumais, 1997) or *Word2Vec* (Mikolov, 2013) – to generate a vector for each headline. We additionally build a large metaphor corpus by running the *Metaphor Magnet* system of Veale (2015) on the Google n-grams (Brants & Franz, 2006), to give millions of metaphors that stretch across diverse topics. Rather than build separate vector spaces for the news and metaphor corpora, we build a *single* vector space for both by appending one corpus onto the other before applying dimension compression. Within this joint VSM, every past metaphor and future headline is assigned a vector of precisely the same dimensionality. It is now a simple matter to measure the angle between the vector for an incoming headline and those of previously encoded *MetaphorMagnet* metaphors.

The metaphor whose vector presents the smallest angle (the largest cosine) to an incoming news vector is chosen as the one with the most relevance to that news item. We built our joint space by compressing 380,000 news items, 210,000 tweets (from sources including @nytimesworld, @CNNbrk and @FOXnews) and 22,846,672 metaphors from *MetaphorMagnet* (which were made available to us on request) into the same LDA space of 100 dimensions. We used the *gensim* package of Řehůřek & Sojka (2010) to build the space, and concatenated word lemmas to their POS tags to provide a richer feature set to the model.

The best pairings produced by this conflation of spaces are tweeted hourly by our bot, called @MetaphorMirror. The thematic basis of the compression means that some pairings show more literal similarity than others, as in:

From @WSJ: *Sultan Abdullah of Pahang has been chosen as Malaysia's new king.*

↑↓

From @MetaphorMirror: *What is a sultan but a ruling crony? What is a crony but a subservient sultan? What drives ruling sultans to be toppled from thrones, appointed by bosses and to become subservient cronies?*

As is evident here, *MetaphorMagnet*'s hardboiled world-view shines through in these pairings, offering meanings and perspectives that, while not actually present in a headline, can be *read into* the headline if one is so inclined. So some pairings show that the VSM has learnt the lessons of history by reading the news, and this shines through too:

From @AP: *Congo's new President Felix Tshisekedi sworn into office; country's first peaceful transfer of power since independence.*

↑↓

From @MetaphorMirror: *How might an elected incumbent become an unelected warlord? What if elected incumbents were to complete tenures, grab power and become unelected warlords.*

Any stereotyping in this response is a product of the VSM and its large corpus of past news, rather than any bias in *MetaphorMagnet*. Though the latter has a symbolic model of warlords and democrats, it is the news space that unites this generic model with the specific history of the Congo. So even as a system strives for topicality, it must have one foot planted in the past if its outputs are to seem informed.

Fifty Shades of Dorian Gray

Much research has been conducted on the analysis of human personality as reflected in our lexical choices. Chung & Pennebaker (2008) describe a tool and a resource, the LIWC (or *Linguistic Inquiry & Word Count*), for estimating personality traits such as *anger*, *affability*, *positivity*, *topicality*, *excitability*, *arrogance*, *analyticity*, *awareness*, *worry*, *anxiety* and *social engagement* from a writer's text outputs. The web version of the tool, *AnalyzeWords.com*, which calculates values for these 11 dimensions by anal-

yzing one’s recent tweets, tells us that *@Oprah* is upbeat and affable as a tweet writer, while *@realDonaldTrump* is upbeat but angry. To create metaphors for a specific person, such as Donald Trump or Oprah, a machine can treat a recent *AnalyzeWords* profile as an 11-dimension vector in personality space, and seek to map this coarse vector to a higher-dimensional space of metaphorical possibilities.

Given the disparity in dimensions between these spaces (11 versus 100) and their different means of construction, we cannot build a joint vector space by just concatenating data. Lacking a dataset to train a neural network to do this mapping across the spaces, we use a symbolic approach to inflate the *AnalyzeWords* space into 100s of dimensions that capture the qualities highlighted in our metaphor set. So we inflate the smaller space by hand-crafting logical formulas – or *transformulas* – to estimate approximately 300 qualities as functions of the eleven core dimensions. Transformulas can conjoin, disjoin and negate these core dimensions. All core dimensions are mapped to the scale 0 to 1.0 (from 0 to 100), and so all transformulas calculate values in the range 0 to 1 also. The negation of a quality simply inverts this scale, so *not angry* can be calculated to be $(1 - \text{angry})$. Consider the transformula for *neurotic*:

$$\text{neurotic}(u) = \text{worried}(u) \times \text{analytic}(u)$$

That is, since neurotics tend to overthink their worries, we estimate the *neuroticism* of user *u* to be the product of the core dimensions *worried* and *analytic*. Likewise, we can say that someone is *narcissistic* to the extent that they are *arrogant* and *self-aware* (given to talking about their own feelings), or *creative* to the extent they are *analytical* and *upbeat*. While transformulas do not reflect an empirical truth about a person, they codify a kind of ‘folk’ symbolic reasoning that lends itself to explicit verbal explanation. Importantly, they allow any Twitter user *u* to be described in terms of the vivid qualities that are used in the NOC list (Veale, 2016) to characterize its gallery of famous people. So, once our transformulas have mapped *AnalyzeWords*’s 11 dimensions into the rich vocabulary of the NOC list, a Twitter user can be compared and matched to its iconic membership. In this way, *@ElonMusk* may show a strong similarity to *Walter White* of *Breaking Bad*, while *@realDonaldTrump* might produce a match to *Lex Luthor*. Such metaphors are a reach – all good metaphors are – but each can be explained in symbolic terms using the logic of the transformulas that link them to their most recent tweets. As such, transformulas turn text-analytic calculations into talking points that a creative linguistic system can exploit.

Consider again the example of *@ElonMusk*, engineer and entrepreneur. From an *AnalyzeWords.com* profile that places his tweets high on the core dimensions *upbeat* and *analytic* and low on the dimensions *angry* and *self-aware*, the transformula qualities *optimistic* ($\text{upbeat} \times \text{analytic}$), *dispassionate* ($\text{analytic} \times \text{not angry}$), *unfeeling* ($\text{analytic} \times \text{not sensory}$) and *determined* (upbeat and not angry) can be inferred. Since three of these transformula qualities – *unfeeling*, *determined* and *dispassionate* – are typical of machines, and the fourth, *optimistic*, is not, our metaphor generator might describe Musk (in light of his most recent

tweets) as an “optimistic machine.” As his *AnalyzeWords* profile also suggests the qualities *laid-back*, *educated* and *scientific*, the latter two of which are typical of researchers, it can also describe Musk as a “laid-back researcher.”

I painted “Optimistic Machine” from *@elonmusk*’s tweets with determined badger-grey, unfeeling Sith-black and dispassionate robot-silver-grey.

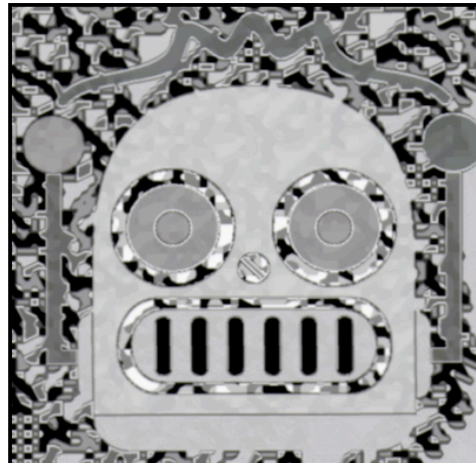


Fig. 1. A personalized metaphor for *@ElonMusk*.

These metaphors, as tweeted by the metaphor-generating bot *@BotOnBotAction*, are shown in Figures 1 and 2.

I made “Laid-back Researcher” from *@elonmusk*’s tweets with scientific *Walter White*, educated priest-black and laid-back *Lebowski-weed-green*.

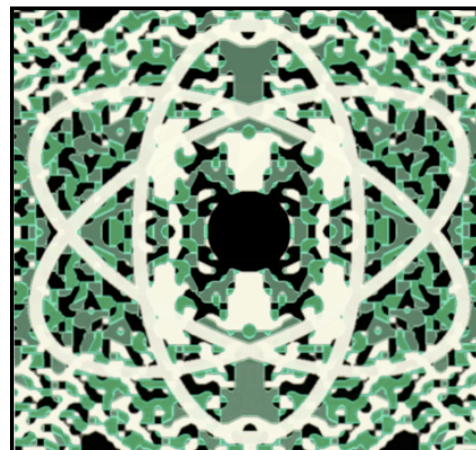


Fig. 2. Alternate personalized metaphor for *@ElonMusk*.

The bot creates a new piece of visual art to complement its metaphors, by creating a 1-dimensional 4-state cellular automaton that unfurls over many rows/generations – and rendering its four states with colours chosen to match the highlighted qualities of the metaphor (see Veale & Cook,

2018). As a flourish, the bitmap of an Emoji annotated with one of the words in the metaphor – so an atom for “scientific” in Fig. 1, and a robot for “robot” in Fig. 2 – is integrated into the image and coloured to suit its new context. Using a colour lexicon in which 600 of the metaphor machine’s stereotypes are mapped to apt RGB codes (e.g., silver-grey for robots, black for priests), it is possible to assign a specific hue to even the non-visual qualities (see Veale & Alnajjar, 2016). Each metaphor is then framed so as to cement this link, so that the black of Fig. 1 is named “educated priest black” while that of Fig. 2 is “unfeeling Sith black.” The tweeted metaphor, comprising four intertwined sub-metaphors, tells us what each of these colours stands for, and suggests how we should feel about them.

It is important to note that *@BotOnBotAction* operates on an *opt-in* basis for most users. The bot will not target them, or metaphorize them, unless explicitly asked to do so with the hashtag *#PaintMySoul*. This kind of personalized computational creativity is not always flattering or welcome, and may even – in some cases – be considered abusive. The exception to this opt-in rule concerns high-profile celebrity users of Twitter, who use the platform to promote themselves to millions of followers. The bot uses the website *TwitterCounter.com* (now defunct) to obtain a list of the most well-known personalities on Twitter, such as Mr. Musk, and freely generates metaphors and images for these luminaries in the downtime between its explicit user commissions. Artists and satirists have always made targets of the powerful and famous, who hardly notice the impudence of a single provocateur; our bot is no different.

Making a Hash of Computational Creativity

Personalization and topicalization offer orthogonal means of grounding the products of CC in the here and now of a user’s reality. Personalization shows that a creative system understands its users, whilst topicalization shows that it appreciates the current and historical context that connects them both. These alternate means of grounding intersect in the task of *recommendation*, for a good recommender engine must understand both the personal dimensions of its users and the topics that matter most to them.

In this section we describe the rationale and the mechanisms of a recommender system for books as embodied in a Twitterbot named *@ReadMeLikeABot*. As with *@BotOnBotAction*, the bot obeys a mostly opt-in policy for its interactions with users, who request ideas for new books to read by using the hashtag *#ReadMeLikeABook*. When the bot is invoked in this way, it uses the text of the invoking tweet as the basis for its recommendation. If this does not offer a foothold to the recommender engine, the bot looks instead at the short Twitter biography that each user defines for their account. If this is empty or unrevealing, the bot finally considers the most recent tweets of the user as a source of topical material for its book suggestions. As in *@BotOnBotAction*, those recent tweets also offer a basis for inferring something of the personality of a user, which may additionally colour the bot’s book recommendations.

The bot also has two activation modes that do not obey a strict opt-in policy. The first is perhaps partially opt-in, insofar as one can request a recommendation for another. In this mode, a user tweets *#ReadHimLikeABook* followed by the Twitter handle of a friend; the bot also accepts the tags *#ReadHerLikeABook* and *#ReadThemLikeABook*. As with *@BotOnBotAction*, the second mode is a filler mode for when the system finds itself between explicit requests. In this case, it exploits the fact that many of the authors in its books database are themselves on Twitter, and so aims to start a conversation about modern literature that draws contemporary writers into an online discussion of books. Authors opt-out of this mode by simply blocking the bot.

Recommender systems are typically either user-based or content-based. In the former, a perceived similarity between users permits a system to recommend items favored by one to the other. In the latter, the similarity function is defined over the items themselves, so a user that favors a given item is likely to favor a similar item too. These two modes are far from orthogonal, as a similarity function for users can be defined over the set of items they both favor, whilst a similarity function for items can be defined over the set of users that favor them both. In short, as a system learns more about its users, it learns more about the items it has in its database of possible recommendations. Importantly, *@ReadMeLikeABot* is not designed to track users, or to learn very much about them, other than that which is public in their Twitter accounts. The bot remembers what it recommends simply to ensure that it does not make the same suggestion again in too short a timeframe. The bot’s user-based recommendations are *personality*-based, while its content-based recommendations are *topic*-based, where each is inferred on the basis of Twitter usage alone.

Recommendation systems are a practical application of AI, yet the task of suggesting existing items permits very little in the way of novelty, no matter how insightful a recommendation may be. Where then lies the computational creativity of a system like *@ReadMeLikeABot*? We view book recommendation not as a creative task in itself, but as an occasion for creativity that allows an expressive CC system to demonstrate a witty and whimsical personality. Consider aspects of linguistic creativity such as metaphor and irony. While a bot like *@MetaphorMagnet* can generate meaningful metaphors with a characteristic voice of its own, its outputs are mostly apropos of nothing, for the bot must rely on its readers to see a serendipitous relevance in its outputs, in whatever context they consume them. Our *@MetaphorMirror* bot finds this relevance for itself in the topicality of the news, yet the bot remains a showcase for metaphorical capability rather than a practical application in its own right. Linguistic creativity is a welcome seasoning for language, rather than the meal itself; it works best when it augments rather than supplants our practical aims. When viewed as a recommender of books, *@ReadMeLikeABot* is not a CC system. Yet the act of suggesting content to a user on the basis of its insights into the user’s personality allows a system to be creative in the expression of its insights, and to find a genuine use for irony and metaphor.

Unauthorized Autobiographies

@ReadMeLikeABot maintains a tiered database of content to recommend. Its first tier contains 500 or so books that are well known, highly regarded, and by authors of some renown. Whenever a book from this tier is recommended, the system can be confident that the user has most likely heard of it, and will likely see its relevance. Each book in this tier is also associated with a set of qualities that describe not just the book itself but the traits of the readers that are most likely to read it. We might assume, then, that philosophical readers enjoy philosophical books. In this way, qualities such as *smart*, *philosophical*, *warm*, *hostile* and *upbeat* can be linked, via appropriate transformulas, to Twitter users who exhibit the same personality traits.

The bot's second tier is much larger, but also much less authoritative. Its 15,000 or so books have been extracted from *DBpedia.org* using the website's SPARQL endpoint. We exploit the linguistic regularity of *DBpedia*'s category terms to also extract a set of themes for every book. When a book is listed under a semantic category with the label *Xs_about_Y* or *Y_in_fiction*, we extract Y as an apt theme. We also mine the hierarchical relations between *DBpedia* categories to build a semantic network that relates these book themes to each other, such as *Artificial Intelligence* to *Neural Networks*. This genre and theme network is then the basis of the bot's content-based recommendations.

The last tier, and certainly the most unusual, comprises 6000 or so humorous fabrications, wholly invented book ideas that wear their artifice on their sleeves. These titles show the usefulness of CC to a recommender system, for when the system has no new content to recommend to a user, it can always fall back on its own in-jokes to fill the gap and keep the user engaged. These inventions must be seen as the literary jokes they are if the bot's credibility as a recommender is not to be diminished in the process. To generate these witty fabrications, we use the *NOC list* of Veale (2016), a large multifaceted database of pop-culture icons that provides vivid descriptions for over 1000 famous people, both real and fictional, modern and historical. For each person, the NOC provides a set of categories – e.g., *billionaire* for Donald Trump, or *politician* for Hillary Clinton – and a set of typical activities, such as *building giant walls* for Trump and *tolerating adultery* for Clinton.

The NOC is a generic, application-neutral resource, but as these examples show, no little humour is baked into the database from the get-go. The NOC list was first built for the WHIM project (the *What-If Machine*), and the task of generating whimsical book ideas can be seen as a what-if scenario: what if Genghis Khan, or Bono, or Tony Stark, wrote a book and told us what they were really thinking? What-if book generation is a simple task using the NOC: a system combines a famous person with an apt category and an associated activity, as in the following examples:

The Comedienne's Guide to Ranting About Liberals
The Rockstar's Guide To Avoiding Taxes
The Son's Guide to Disappointing the Family

These faux books are credited to, respectively, Roseanne

Barr, Bono, and Fredo Corleone. When the NOC entity is fictional and has a known creator, this information is also used in the generation of literary what-ifs. Consider these:

Captain Ahab's Guide To Chasing a Great White Whale
Dr. Stephen Strange's Guide to Performing Magic Tricks
Yoda's Guide to Promoting Mysticism

These books are credited to Herman Melville, Stan Lee and George Lucas, respectively. What-ifs also give us the opportunity to imagine incongruous pairings of authors:

The Geek's Guide to Studying Science
The Psychiatrist's Guide to Probing the Mind
The CEO's guide to Pioneering New Technologies

The first is credited to Peter Parker and Wesley Crusher; the second to Drs. Sigmund Freud and Frasier Crane; the third to Tony Stark and Steve Jobs. In general, any linguistic framing of pop-culture factoids that pokes fun at the book industry will suffice here. Publishers themselves see the value of parodic cash-grabs, and shelves already groan under fictive offerings like the following, by Pablo Escobar, Tyrion Lannister and Wile E. Coyote, respectively:

Lifting The Lid on The Medellín Cartel
Exposed: The Secrets of The House Lannister
An Insider's Guide to A.C.M.E.

Recall that such non-books are only ever recommended to the user when better matches from a higher tier cannot be found, or when all have been offered to that user already. Their value is largely found in repetition, then: the more a user interacts with the bot, the further down its tiers the bot must descend, and more the bot will reveal its sense of humour, about books and about the book industry itself.

They Shall Not Grow Bold

Much research has focused on the recognition of sarcasm and irony in text, especially as it is used in social media. This emphasis on detection is not surprising, given that so much of the language that matters is created by humans. In contrast, very little research has addressed the creative task of *generating* irony and sarcasm, no doubt because we already find our machines to be inscrutable enough in their dealings with humans. But more than that, sarcasm and irony cannot exist outside of a specific communicative goal: we can generate metaphors in a null context and leave it to the reader to unearth their implied meaning, but irony and sarcasm require a firm context to push against. In short, they need realistic expectations to bring to bear, and a context that undermines them in ways for all to see. For a machine to generate irony and sarcasm well, it must be given enough of these expectations to be versatile, and an ability to identify those contexts that clearly fall short.

Personality-driven recommendation supplies these expectations in convenient qualitative and quantitative forms. When the bot has a topic-based reason to recommend e.g., an intellectual book to a reader who scores low on the analytic dimension, or is poorly scored by the transformula for *intellectual*, this mismatch between topic and person-

ality is just the failure of expectation that irony demands. In this case, topic-based recommendation creates the expectation, and personality analysis defies it, giving the bot a logical reason to snarkily poke fun at the disparity. Suppose the bot does suggest an intellectual book, on cosmology, say, to a user with an avowed interest in cosmology that appears to fall well short of the intellectual bar; how should it wittily allude to this failure of expectations? The bot can learn from how humans deal with disappointment by looking to how we express our dissatisfaction through irony. So a web search for *intellectual* finds the following ironic similes: *about as intellectual as a Cheez Doodle, as a cucumber, as a brush, a hole in the ground, a wart hog, a potted plant, a bulldog, an emu*, and others too rude to repeat here. What links each of these mental images is not a shared feature but a common framing; in each case, the author prefaces a simile with “about” to signify the semantic imprecision of a creative liberty. We can exploit this framing device to seek out many other ironic similes on the Web for any quality one cares to undermine, to give a bot a rich palette of ironic options to use on its own users.

When a user’s Twitter profile scores low for a quality that is estimated either directly (using *AnalyzeWords*) or indirectly (via a transformula), the bot will dip into its bag of ironic similes for that quality. Choosing at random, it can frame what it retrieves in a variety of colourful ways. Suppose the quality is *philosophical*, and the bot retrieves the simile *about as philosophical as a bowel movement*. Perhaps the bot also intends to recommend the philosophical novel *Steppenwolfe* by Hermann Hesse, as the user’s recent tweets mention *loneliness* and *alone*. It can frame this pairing of a book to a simile in the following ways:

Hey @bookreader, if you're as philosophical as a bowel movement then maybe you should read 'Steppenwolfe' by Hermann Hesse on the theme of solitude.

Hey @bookreader, I used to be as philosophical as a bowel movement until I read 'Steppenwolfe' by Hermann Hesse on the solitude theme.

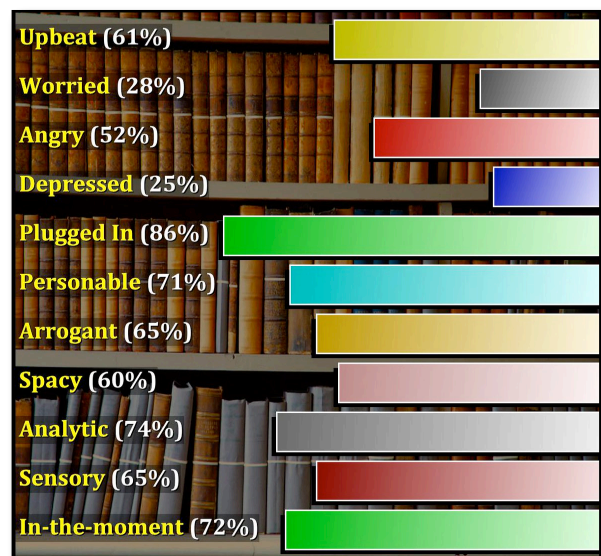
Hey @bookreader, given your personality profile I don't know which philosophical book is more you: 'Steppenwolfe' by Hermann Hesse on the solitude theme, or 'The Bowel Movement' by Stephen Tolkein.

The first framing was used in early field tests of the bot, in its prelaunch in the weeks before the 2018 *Science and Communication* conference (for which the bot was commissioned). As might be expected, its in-your-face humour was not popular with everyone, and was a cause for some dismay to the event’s organizers. The “If you’re X” construction did little to salve the pain of a sudden insult from an abusive bot, even if the user had invoked it explicitly. The second framing proved to be more successful, since it now turned the bot’s humour inward, on itself, rather than outward on users who might see themselves as its victims. The third framing turns it outward again, on the user, but in a more subtle guise that presents it not as a direct insult but as a playful joke at the expense of the book industry.

Note how the bot is forced to invent an author for its literary in-joke, which it does by cutting up the author names from its first tier of books. The third framing is especially apt when the bot’s tweet is accompanied by a graph of the user’s 11-dimension personality profile (see below), since it allows one to appreciate the basis for the bot’s response.

But the second framing has another advantage, in that it allows the bot to speak directly to the topic of the recommended book. Consider this particular response to a user:

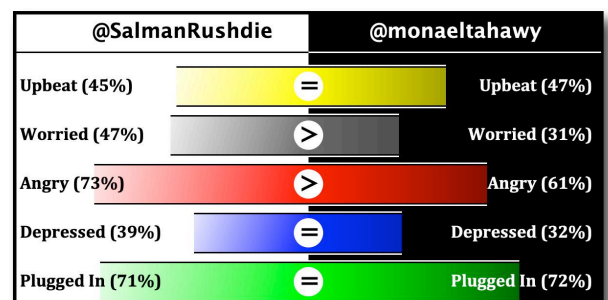
On the prettiness theme, @anonymized, I used to be as attractive as a brown cardigan until I read "The Picture of Dorian Gray" by Oscar Wilde. How about you? I crunched your recent tweets:



When the bot is between user requests, it attempts to start a conversation about books and their ideas. It does so by posing literary questions to its readers, as in this tweet:

On the religion theme, which of these books is more provocative than the other? "The Satanic Verses" by @SalmanRushdie, or "Headscarves and Hymens: Why the Middle East Needs a Sexual Revolution" by @MonaEltahawy? I compared their recent tweets.

The question itself typically provokes much less conversation on Twitter than the side-by-side personality analyses that the bot provides of the authors’ most recent tweets.



As You Like It: The Question of Evaluation

A Twitterbot that publicizes a user’s psychological profile is rather like a public *speak-your-weight* machine. No one likes to be judged, least of all by a computer, and we can expect a wide diversity of views on the use of tools like *AnalyzeWords* to condition a bot’s creative outputs. These range from “Awesome!” to “very creepy,” with the rump of users taking their analyses as a starting point for further wit of their own. One user replied to the book bot’s ironic confession ‘I used to be as poetic as a donkey passing wind until I read “Romeo and Juliet” by William Shakespeare’ with the wry remark “I’ll have you know that I’m still as poetic as a donkey.” Another user, for whom the bot recommended Alex Comfort’s “The Joy of Sex” (on the *love* theme) replied “Wow wow, easy there bot ... buy me dinner first!” The author @MaggieEllen replied to the bot’s comparative analysis of herself to @AmyTan with a trope from *The Simpsons*, “you my overlord now?” before addressing the specifics of the analysis with the remark “Just real glad to know @AmyTan and I are both on 300 mg extended release Welbutrin and equally depressed.” The value of a creative agent lies as much in the creativity it fosters in others as in the creativity of its own outputs.

That said, users are more open to personalized outputs when they flatter their targets, and often chafe at the negative aspects of analyses that are otherwise quite positive. One famous comedian with a science background, whose Twitter feed reflects his TV presence – half comedy, half science – was described by @botOnBotAction as “the best of Peter Parker and the worst of Jim Carrey: scientific and intelligent yet cloying and insecure.” The user’s response was unforgiving: “Not so insecure that I post anonymously though.” When a science book by the same user, a popular author, was promoted by the book bot via its *AnalyzeWords* comparison to a similar author, the mention earned it a comparable rebuke “Not sure this analysis has a single thing to do with the books; but you enjoy yourself.” When creative systems get personal, so too will their audiences, making it difficult to objectively evaluate their outputs.

This makes an extrinsic evaluation of such systems preferable. Ghosh & Veale (2017) explore the contribution of a user’s Twitter profile – specifically, their *AnalyzeWords* profile – to the assessment of whether their most recent tweet is sarcastic or not. We expect mood and personality to be factors in the determination of a sarcastic mindset, as recent emotions – from anger to arrogance – will shape the perception of a user’s intent in a given tweet. In that case, we expect a neural model of sarcasm detection to be improved by the addition of accurate personality features that are active in the relevant time frame. Ghosh & Veale report a statistically significant gain in detection accuracy, of approximately 6% to 7%, when *AnalyzeWords* features are incorporated into their neural architecture. If personality features can improve the appreciation of creative texts, they can certainly play a key role in their generation too.

Topicality-driven bots like @MetaphorMirror afford a more direct evaluation, since it is the news context, and not a specific user, that is directly addressed in the output.

Vector Space	Low	Avg.	Good	V.Good
LDA stories+tweets	1.1%	47.8%	41.1%	10%
LDA stories only	3.3%	65.6%	30%	1.1%
LSA stories+tweets	10%	60%	30%	0%
LSA stories only	17.8%	64.4%	16.7%	1.1%
Word2Vec	10%	57.8%	32.2%	0%
Random baseline	45.5%	46.7%	6.7%	1.1%

Table 1. Distribution of Aptness by choice of model.

We used *CrowdFlower* (now *Figure-Eight*) to elicit human ratings for 90 metaphor / headline pairs from different models. A scale of 1 ... 5 was used for ratings on three dimensions: *aptness*, *comprehensibility*, and *influence*, the last of which marks the extent to which a metaphor shapes a rater’s response to a headline. Six different models were used to select the ‘best’ metaphor for each of the 90 headlines: an LDA topic model built with a corpus of 380k news stories and 210k news tweets; an LDA model built with the 380k news stories but no tweets; an LSA model built with 380k stories and 210k tweets; an LSA model built with 380k stories but no tweets; a Word2Vec space of pretrained vectors, so no news stories or tweets were used; and a baseline model that pairs a random metaphor to each headline. For each variant of the LDA and LSA models, 22.84M *MetaphorMagnet* metaphors were concatenated to the news content (stories with/without tweets), so these models produced joint *news + metaphor* spaces.

Mean ratings for each dimension in different models were not very discriminating. In each case, LDA (stories + tweets) pipped all others to the top spot. For *Aptness* – how apt is a metaphor for a headline? – the means ranged from 2.95 (\pm standard dev. 1.27) for LDA (stories+tweets) down to 2.20 \pm 1.2 (random baseline). *Comprehensibility* – the understandability of each pairing – ranged from 3.59 \pm 1.05 (for LDA, stories+tweets) down to 2.54 \pm 1.12 (random baseline), and *Influence* ranged from 3.01 \pm 1.24 (LDA stories+tweets) to 2.09 \pm 1.24 (random baseline). The differences across models were not statistically significant, except in comparison to the baseline. Yet mean performance disguises deeper differences. If we quantize the human ratings of aptness into four equal-sized buckets (*Low*, *Average*, *Good* and *Very Good*) so as to identify the model that places the most metaphor:headline pairs into the *Good* or *Very Good* buckets, we obtain the findings of Table 1. More than half of pairings suggested by the LDA (stories+tweets) model end up in one of these top buckets, suggesting that this model produces the most apt results.

Conclusions: Don’t Give Up The Day Job

Oscar Wilde once wrote that “art has as many meanings as man has moods.” The point of affective computational creativity is not just to enlarge the space of artifacts that is explored by a CC system, or to make those artifacts more revealing about the processes that generated them; it is to make those artifacts more revealing about their *audiences*.

This potential for personalization and topicalization has not gone unnoticed in other CC work. With regard to *The Painting Fool*, a versatile generator of portraits (and other painterly forms), Colton *et al.* (2007) built on the work of Pantic & Rothkrantz (2003) to give the Fool a sense of the mood of the subject it is painting, so that it might capture this understanding in its outputs. A linguistic tool such as *AnalyzeWords* is of little use when dealing with a video or a camera still, but Colton *et al.* note the value of FACS, the *Facial Action Coding System* of Ekman (2002). Some users of CC systems wear their emotions on their faces; others reflect them in their social-media communications. Depending on the modality of the interaction – and personality certainly turns CC into an interactive process, even if users are scarcely aware of their own contributions – a system must exploit whatever affective cues it can find. Topicalization has also revealed a strong potential for CC exploitation. Like personalization, it makes the outputs of a generative system more relevant to the users for whom they are created. For example, the *PoeTryMe* poetry generator of Gonçalo Oliveira (2017) augments its core knowledge stores (such as semantic and conceptual networks) in a number of ways, including the use of live Twitter feeds to ground its outputs in the here-and-now of social media. By showing an awareness of users and their world, these systems present themselves as more self-aware too. They present themselves not as closed generative bubbles, like the imprisoned wretch of Searle’s *Chinese Room* thought experiment (1980), but as agents of a wider world that can predict how their creative outputs will impinge on others.

If viewed as ‘human’ creators, CC systems such as *The Painting Fool*, *PoeTryMe* and *MetaphorMagnet* would all be seen as full-time creators whose work is their calling. Most CC systems conform to this all-or-nothing pattern; their creative work is everything, and the systems have no ‘lives,’ whatever this might mean, beyond their generative responsibilities. *@ReadMeLikeABot* is a useful exception to this generalization. To this CC system, as it is to most humans, creativity is merely a sideline to a ‘day job’ that is not in itself a creative exercise. Book recommendation is a task that requires AI but has little obvious use for CC, yet this bot shows that a system that benefits from an appreciation of a topical context, or a user’s personality, can also reap benefits from the creative framing of its outputs. Conversely, the CC component of these systems may also benefit from exposure to the stuff of its mundane day job, by giving it a contingent knowledge of possibilities that it might never recognize in a purely creative mode. We can go further, and argue that *all* CC systems can benefit from a day job that exposes them to the mundane concerns of the people they must serve, so as to later transmute those concerns into something both familiar and non-obvious.

References

- Blei, D. M., Ng, A. Y. & Jordan, M. I. (2003). Latent Dirichlet Allocation. *J. of Machine Learning Research*, 3:993–1022.
- Brants, T. & Franz, A. (2006). *Web IT 5-gram Version 1*. Linguistic Data Consortium.
- Chung, C.K. & Pennebaker, J.W. (2008). Revealing dimensions of thinking in open-ended self-descriptions: An automated meaning extraction method for natural language. *Journal of Research in Personality*, 46, 96–132.
- Colton, S., Valstar, M.F., & Pantic, M. (2008). Emotionally Aware Automated Portrait Painting. In Proc. of DIMEA’08, the 3rd Int. Conf. on Digital Interactive Media in Entertainment and Arts Athens, Greece. September 10 - 12.
- Ekman, P. (2002). *Facial Action Coding System. A Human Face*. Utah, Salt Lake City: W. V. Friesen, and J. C. Hager.
- Fauconnier, G. & Turner, T. (2002). *The Way We Think: Conceptual Blending and the Mind’s Hidden Complexities*. Basic Books, New York.
- Ghosh, A. & Veale, T. (2017). Magnets for Sarcasm: Making Sarcasm Detection Timely, Contextual and Very Personal. *Proc. of EMNLP 2017, the Conf. on Empirical Methods in Natural Language Processing*, 493–502, Copenhagen, September 7–11.
- Gonçalo Oliveira, H. (2017). O Poeta Artificial 2.0: Increasing meaningfulness in a poetry generation Twitter bot. In *Proc. of CC-NLG, the Workshop on Computational Creativity in Natural Language Generation*. Santiago de Compostela, Spain pp 11-20.
- Koestler, A. (1964). *The Act of Creation*. Penguin Books.
- Lakoff, G. & Johnson, M. (1980). *We Live By*. University of Chicago Press.
- Landauer, T.K. & Dumais, S. (1997). A solution to Plato’s problem: the latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psych.Review* 104(2):211-240.
- Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781 [CS]*, January.
- Pantic, M. & Rothkrantz, L.J.M. (2003). Toward an affect-sensitive multimodal human-computer interaction. In Proc. of the IEEE, 91(9):1370–1390.
- Řehůřek, R. & Sojka, P. (2010). Software Framework for Topic Modeling with Large Corpora. In Proc. of the LREC 2010 Workshop on New Challenges for NLP Frameworks, pp 45–50.
- Searle, J. (1980). Minds, Brains and Programs. *Behavioral and Brain Sciences* 3(3): 417–457.
- Veale, T. (2016). Round Up The Usual Suspects: Knowledge-Based Metaphor Generation. Proc. of the *Meta4NLP Workshop on Metaphor at NAACL-2016, the annual meeting of the North American Assoc. for Computational Linguistics CA: San Diego*.
- Veale, T. (2015). Unnatural Selection: Seeing Human Intelligence in Artificial Creations. *Journal of General Artificial Intelligence*, 6(1), special issue on Computational Creativity, Concept Invention, and General Intelligence, pp 5-20.
- Veale, T. & Alnajjar, K. (2016). Grounded for life: creative symbol-grounding for lexical invention. *Connection Science*, 28(2):139–154.
- Veale, T. & Cook, M. (2018). *Twitterbots: Making Machines that Make Meaning*. Cambridge, MA: MIT Press.

Computational Analysis of Content in Fine Art Paintings

Diana Kim

Computer Science
Rutgers University
New Brunswick, NJ, U.S.A
dsk101@rutgers.edu

Jason Xu

Operations Research and Financial Engineering
Princeton University
Princeton, NJ, U.S.A
jasonx@princeton.edu

Ahmed Elgammal

Computer Science
Rutgers University
New Brunswick, NJ, U.S.A
elgammal@cs.rutgers.edu

Marian Mazzone

Art and Architectural History
College of Charleston
Charleston, SC, U.S.A
marian.mazzone@gmail.com

Abstract

We propose a deep learning algorithm that can detect content and discover co-occurring patterns of the content in fine art paintings. The following intellectual merits are the motivations of our project.

First, the content detection provides a baseline of Computational Iconography (CI), which is to understand what objects/subjects can be seen in fine art paintings. Second, we argue that the found co-occurring patterns chart meaningful connectivity across content in art. Third, we imbed our system in Computational Creativity (CC) in a broad sense. By the nature of our system of machine learning, it creates informative connections between different modalities (images/words), which are not initially constructed or intentionally specified. Our system is automatically trained to discover the connective patterns reflecting artists' creativity, which are latent in the large dataset of paintings.

To build a content detector, we adopted an Inception-V3 (ImageNet) and fine-tuned it over 40,000 paintings with the words extracted from their titles. We validate that our system detects content information fairly (68% precision rate at the top content). Also, we find that the last fully connected layer parameters of Inception-V3 are trained to encode general co-occurring patterns between content. We validate that the co-occurrence can be interpreted as relatedness among content in art.

Introduction

In this paper, we present a computational method that can understand the content of fine art paintings. By bringing our problem on the broader stance of general art, we highlight our system interprets art, especially in terms of the content, which is one of the three principles for understanding art: form, content, and context (Dyke 1887; Lowry 1967).

More specifically, we adopt a deep learning approach and argue how it automatically creates many virtual connections from a target painting to the multiple relevant pieces of information called content: the objects, activity, or other information that can be seen in the painting. First, we implement a content detector to connect a fine art painting (image) and relevant output words (content). It creates plenty of textual information about a given visual entity. Second, as a by-product of the content detector, we find that distributed vector representations, of mutual distances capture the general

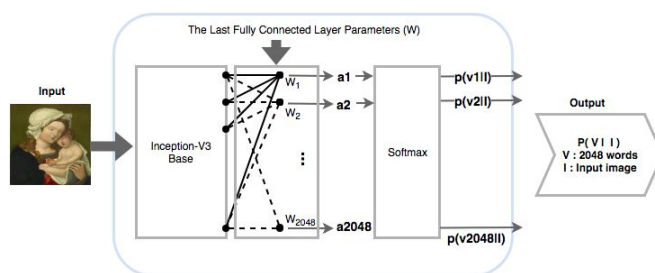


Figure 1: Content Detector

co-occurrence patterns among content. We prove that the co-occurrence of content can be interpreted as the relevance among content in art, which is embedded in large training set of paintings.

Our proposal for the computational content analysis can have the following practical and intellectual merits. The found virtual connections would be useful to associate words to the images/words we are focused on, so it drives us to other resources flowing into other relevant paintings. It enables us to reach more paintings from a few key words of general content. This suggests a feasible application of our system to improve general accessibility to digital art retrieval systems. Currently, art retrieval systems require highly specialized knowledge such as title, author, genre, time period, or style of paintings, which ordinary users may not know well.

Furthermore, building computational models for understanding human creative products can be a fundamental part of the field of Computational Creativity (CC). We argue our computational model links to broad perspectives of CC. Although our methodology does not precisely articulate how artists' mind operates on their creative artifacts or create novel products, it does focus on artworks which are objects of human creation, and it may give us insight into the pattern of connections among concepts, words, and visuals that artists use when making their images.

Our machine takes in many paintings as inputs and learns to connect images and words as a reflection of how the input artifacts are presenting. The connections are not pre-constructed or designed by the authors or from any external knowledge of art. They are instead solely the result of the

huge processing capacity of the machine to work with images and words. We believe that by analyzing such a large number of paintings, the machine is able to reflect the associative patterns of images and co-occurring concepts that human artists may be using when they create their artworks. Our system may be related to the broad definition of CC (Bown 2012), in that a computationally creative system is not necessarily modeled on the human mind or on human goals, but does apply to the occurrence of creation.

In previous computational art analysis, most research works have focused on visual appearance and its descriptions, i.e., visual forms, such as brush strokes (Elgammal, Kang, and Den Leeuw 2018; Hendriks and Hughes 2009) and stylistic analysis (Kim et al. 2018; Elgammal et al. 2018). However, as we consider three art principles, which are primary elements for understanding art (Lowry 1967), analysis grounded purely in visual forms may not be a sufficient approach. We can better appreciate art if we understand content, including the subject matter and historical context of interpreting that content. In art history, this approach is called the study of iconography and iconology, with its most notable practitioner being Erwin Panofsky (1892–1968). Hence, we devise a computational framework for content and it provides the baseline work for Computational Iconography (CI).

To build our content detector, we adopt and fine-tune a deep neural network architecture, Inception-V3 (Szegedy et al. 2016), for which the input is an image (painting) and the output is a probability mass function (pmf) as shown in Figure 1. In the model, the pmf’s support domain V is 2,048 words, so through probabilistic representation, we can quantify the relevance of each word to an input image I . For fine-tuning, we only re-train parameters of the last fully connected (FC) layer and other parameters are transferred from a pre-trained ImageNet. While the training proceeds, we observe an interesting property: the network starts learning to capture associative patterns between the output words. For more details, (W_i) in Figure 1, the weight vectors are trained to be a distributed representation set, i.e., their mutual distances can encode certain relationships between the content (words) in paintings. Although we intentionally train the machine to create linkages between an image and words, but the machine also autonomously learns to capture relationships among words, too.

We can observe the following features.

- Words denote concepts that are visually similar from the perspective of the machine, if (and only if) the word representations are likely to be close each other.
- Concepts often co-occur within a painting, if (and only if) the corresponding word representations are likely to be close each other.

From the above analysis, we can notice distinct characteristics of our vector representation through differentiation with the word embedding systems in Natural Language Processing (NLP). In NLP, word embedding models (Mikolov et al. 2013) encode syntactic or semantic similarities between words through the context of the likeness to their neighboring words. On the other hand, our embedding sys-

tem encodes word relationships based on the visual similarities or co-occurring patterns of the concepts over a whole set of fine art paintings, i.e., instead of adjacent words, paintings become major contextual resources to extract relationship between words.

We specially point out that the second co-occurrence property could offer us clues about which subjects and concepts typically occur simultaneously in paintings. Using this, we could easily find repeated motives across paintings. In practice, it is one of main tasks of iconography, but may not be easy if we only have the pure semantics of the words available, or if we look for them only through the human naked eyes. For instance, in our distance analysis, the two words ‘Virgin’ and ‘Angel’ are the closest words. Their pure semantics are not highly correlated, but we confirm that they are two primary co-occurring components for the subject of ‘Annunciation’. It is a popular theme in paintings during the Middle Ages and Renaissance.

The method may not be able to find immense and delicate symbolic meanings as art historians have done, such as Erwin Panofsky’s discovery of a connection between lilies and Mary as a symbol of her chastity in *Mérod Altarpiece* (Panofsky 1971). However, clues are sometimes enough to initiate deeper directed studies, especially when we deal with the massive archive of paintings. Furthermore, we also know a fact: iconographic analysis should begin with the object that can be seen from the art works (Munsterberg 2009).

In summary, we claim the followings.

1. Our system detects content information fairly well. As the system is designed to detect multiple labels, the loss objective in training does not measure actual performance well. We validate the performance through the following alternative methods: (1) Comparison between machine pmfs and words populations. (2) Human subjects survey with students in art history.
2. Our system discovers co-occurring patterns and it implies certain relatedness among content in art. We validate the claim through the correlation analysis between the degree of co-occurrence (mutual distances between the vector representations for two key words) and the relevance (number of results to searching queries of intersecting two key words).

In the following sections, we will explain the whole procedure of implementing a content detector and achieving distributed representations. We will also explain evaluation procedures and its results. In the last discussion section, we will draw a practical application of our system on current digital art searching platforms.

Related Works

Our problem shares a common goal with some prior researches about computational content analysis in art collections (Carneiro 2011; Carneiro et al. 2012; Crowley and Zisserman 2014; Picard, Gosselin, and Gaspard 2015).

To our best knowledge, Gustavo (2011) (Carneiro 2011)’s graph-based learning algorithm was the first computational approach to detect content in art works. He annotated digital art prints with 28 pre-defined semantic labels. Before

his work, most computational art analysis had focused on visual forms such as brush strokes (Polatkan et al. 2009; Hendriks and Hughes 2009) or stylistic analysis (Jafarpour et al. 2009).

Later, Gustavo *et al.* (Carneiro et al. 2012) proposed other computational approaches (random, bag of features, label propagation, and inverted label propagation) to detect 75 content classes from monotonic paintings. By dividing the targeted annotations into global semantic, local composition, and local pose, they tried to detect more structured semantics from the more general paintings than their previous works.

Elliot *et al.* (Crowley and Zisserman 2014) used a transfer learning scheme. They showed that object classifiers trained by natural images can effectively detect objects in paintings. They compared the performances of two Support Vector Machine (SVM) classifiers, in which each machine is trained with one of two feature sets: Fisher vector representations or vector collections from an intermediate layer of the Convolutional Neural Network (CNN). In the result, the CNN outperformed the Fisher vector representation. However, their experiments were limited to object-oriented concepts, such as chair, bird, and boat, and there were only 10 classes.

David *et al.* (Picard, Gosselin, and Gaspard 2015) used the same methodologies as the work of Elliot *et al.* (Crowley and Zisserman 2014), but applied them to annotate cultural heritage collections. During the experiment, they classified artifacts in one of 459 semantic classes. Differently to the result of Elliot *et al.* (Crowley and Zisserman 2014), the performance of the Fisher vector representation was slightly better than one of the CNN features.

In our methodology, we applied the deep-learning method to understand the content in fine art paintings and validated its performance. For the data set, we did not use any pre-defined words like those of previous works. Instead, we collected words from the titles of paintings and selected 2,048 words based on the words' statistics. Along with content detection, we also found associative patterns (co-occurring or visually similar) between the content in paintings.

Methodology

Our primary goal is to design a system that can represent a conditional pmf: $P(V|I)$, where V represents a word whose domain is a 2,048 words set and I is an input image. Based on probability, we will try associating highly probable words with an input image and validate their association.

Architecture

To design the probabilistic system, we utilize a multiple labeling training by modifying an original machine learning algorithm, Inception-V3. We train the same network architecture by using its original objective function. However, as the original algorithm can handle only multiple class problems (class labels are mutually exclusive) setting only one class as probability one, we have to change the framework to enable it to carry multiple non-zero probabilities. For a K multi-class problem, the network's output produces a pmf whose k -th element implies the probability of the k -th word

(v_k) given an input image I . The original objective function is a softmax cross-entropy for training data samples. Let a_k be the k -th value before the softmax layer in the network of Figure 1. Then the output probability P_k and the objective function E of N samples are the following.

$$P_k = P(V = v_k|I) = \frac{\exp^{a_k}}{\sum_k^K \exp^{a_k}} \quad (1)$$

$$E = - \sum_n^N \sum_k^K I_{k,n} \cdot \log_e(P_k)$$

In equation 1, $I_{k,n}$ is an indicator function stating whether or not the n -th sample belongs to class k . In our project, to handle the multiple labels, we re-define an objective function E' with a $J_{k,n}$ instead of the indicator function.

$$P_k = P(V = v_k|I) = \frac{\exp^{a_k}}{\sum_k^K \exp^{a_k}} \quad (2)$$

$$E' = - \sum_n^N \sum_k^K J_{k,n} \cdot \log_e(P_k)$$

In equation 2, the summation of $J_{k,n}$ over all k is equal to one ($\sum_k^K J_{k,n} = 1$), and each value is $J_{k,n} = \frac{1}{L}$, if the n -th input image has a k -th word and the total number of labels of the image is L . We can interpret the E' as a negative log likelihood function if we draw a case in which multiple words for each sample image are independently generated by the P_k . Suppose we have three labels (v_1, v_2, v_3) for an image, then a $P(v_1, v_2, v_3|I)$ equals $\prod_{k=1}^3 (P_k)$. Then we can compute the $E' = -\frac{1}{3} \sum_{k=1}^3 \log_e(P_k)$ for the sample image and its labels. If we consider each pair (I, v_1) , (I, v_2) , (I, v_3) as independent samples, then it is the same as the original multiple class objective function E except for the normalization factor of $\frac{1}{3}$. Internally, we use multi-class training L times and compensate its multiple uses by dividing it by L . In this sense, the modification does not harm the primary concept of cross entropy that the original algorithm intends and it can handle multiple label training.

The Last Fully Connected Layer Weights

In NLP, a skip-gram model (Mikolov et al. 2013) can learn distributed representations of words by capturing statistical patterns with their neighboring words in a text corpus. If two words' neighboring words are often similar, their representation also become close. Inspired by the idea, we hypothesize that the last layer weights of our network can also encode an associative relationship between the 2,048 output words. If two images are visually similar but labeled by two different words, then the two word representations are expected to be close.

We can think of two cases in which images are the same or visually similar, but labeled by different words. For the first case, in general, low-level concepts are visually similar if they have a common ancestor in the concepts' hierarchical system. For example, specific kinds of flowers such as lanaculus, rose, or camellia are necessarily mapped into very

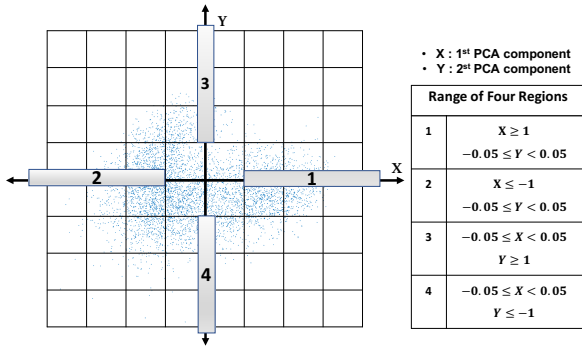


Figure 2: Four Regions in two components of PCA space (Blue dots: 2-D PCA transformed points of validation data)

close points in the top hidden layer of a neural network layer, but each of the points are to be labeled with different names.

For the second case, if some concepts often co-occur in paintings, it corresponds to the case in which the painting images are the same but labeled by different concepts. For instance, Christ, cross, angel, and a subject of lamentation are often delineated in a painting. Similarly, Madonna, child, and Saints often co-occur, too.

In this context, we examine the last weights W (2,048 x 2,048) in Figure 1 as distributed representations for the 2,048 concepts, and confirm that they are close to one another if (and only if) their corresponding concepts are visually similar or frequently co-occurring in paintings.

Validation Methods

In this project, we have two claims. One is we can build a probabilistic system that can have higher probabilities on words more relevant to an input painting. The second one is that parameters (words distributed representations) collected from the last layer of the system can encode the relationships between the words in fine art paintings. In the following sections, we will explain how we have validated the claims.

Content Detection To validate the first claim, we conducted a survey to determine how many subjects agree with the machine’s 10 most probable words as relevant concepts. More detailed survey results and its steps are presented in the later survey section in the Experimental Result. As the second evaluation method, we performed the following experiment: we collected image embeddings from the hidden layer right after Inception-v3 base (in Figure 1) by inputting training images. Then, we learned two Principal Components Analysis (PCA) components (occupying 10% of the total variance of the embeddings). In the found PCA space, images near the points of $[c, 0]$, $[-c, 0]$, $[0, c]$, and $[0, -c]$ (in our experiment, c is 4) showed a certain degree of consistency in their content, so we set the following hypothesis and validated it.

If a machine can detect content from an input painting well, then the following two statistics will be similar to each other. One is the sample frequency histograms of

Group	Ranking Ranges
Group 0	One word ranked 1 (the word itself)
Group 1	Ten words ranked 2-11
Group 2	Ten words ranked 1025-1034
Group 3	Ten words ranked 2039-2048

title-words of four groups of images, located near the points $[c, 0]$, $[-c, 0]$, $[0, c]$, and $[0, -c]$. Each group of images are the validation images that are PCA transformed to the regions defined by the ranges of 1, 2, 3, and 4 in Figure 2. Another statistic is the machine’s output pmfs, as we individually pass four simulated image embeddings into the network after the Inception-V3 base (in Figure 1). Each of the four simulated image embeddings has been computed by an approximate inverse PCA on the vectors $[c, 0]$, $[-c, 0]$, $[0, c]$, and $[0, -c]$.

Let d be the number of PCA components, s the number of validation samples, H the collected sample embeddings from the hidden layer, and g the size of the dimension of the hidden layer. As we do whitening PCA on the hidden layer embedding H ($g \times s$), a PCA transformed T ($d \times s$) can be written as

$$T = \Lambda^{-\frac{1}{2}} \cdot \Theta^T (H - m) \quad (3)$$

where m is a mean vector computed from H , Θ is a matrix ($g \times d$), whose columns are orthonormal vectors to define the PCA’s principal axes, and Λ ($d \times d$) is a diagonal matrix defining the PCA variances. By using 3, we can simulate the four embeddings \hat{h}_z ($g \times 1$) that equal $\Theta \cdot \Lambda^{\frac{1}{2}} \cdot t_z + m$, where $t_1 = [c, 0]^t$, $t_2 = [-c, 0]^t$, $t_3 = [0, c]^t$, and $t_4 = [0, -c]^t$, z is in $[1, 4]$. Now, we can compute the machine’s output distribution \hat{y}_z in 4 and the W' has the same columns of W except for the last bias column vector w_{bias} . The \hat{y}_z is the network outcome when inputting the simulated embedding \hat{h}_z into the last FC layer in Figure 1.

$$\hat{y}_z = \text{softmax}(W' \cdot \hat{h}_z + w_{bias}) \quad (4)$$

Words Distributed Representation After finishing training, we collected the last layer parameter W in Figure 1, and regarded each of the i -th rows (w_i) as the distributed representation of the i -th word. We computed cosine similarities between the representations and formed a matrix M in 5. Each component $M_{i,j}$ represents the cosine similarity between the representations of the i -th and the j -th word.

$$M_{i,j} = \frac{w_i^t \cdot w_j}{|w_i| \cdot |w_j|} \quad \forall i, j \in [1, 2048] \quad (5)$$

To find relationships between distance, we sorted each row of M in descending order and for each row, we set the first word as group 0 and collected the other three groups of words according to their rankings as shown in Table 1.

To verify that closer words in the distributed representations are more correlated words in art, we tried searching artworks in GoogleArt&Culture

(<https://artsandculture.google.com>) with queries of intersecting two words. Its first word is the group 0 word and another word is from group 1, group 2, or group 3. We posit that returning more results as we query an intersection of two words is a reflection of more connections between the words in the art domain. GoogleArt&Culture searches artworks by intersecting all input words and matching them to words in the documents in its database, which have basic information (author, title, and year) or general descriptions about paintings. Hence, the number of retrieved art works should generally decrease with successive groups 1, 2, and 3 if the distributed representations can encode correlations between words within art.

Experiment Results

Data set – Paintings and words from titles

We used a public collection of fine art paintings, the WikiArt (<https://www.wikiart.org>) data set. The collection has more than 60,000 paintings covering the Renaissance to the Modern period. Instead of using all of them, we utilized paintings drawn before the 20th-century (50,160 images) and split them into ‘Train’ (85%), ‘Validation’ (10%), and ‘Test’ (5%). We used ‘Train’ in training the Inception-V3 and defining PCA’s principal axes, ‘Validation’ for evaluations and a survey, and ‘Test’ for presenting test results.

To prepare training samples, we labeled the paintings with words from each painting’s title. All words from the titles are good sources for understanding the content of target paintings, but we do not want to use words that appear too sparsely or refer to specific entities, such as the name of an area or a person. Using the Natural Language Toolkit (NLTK) library (version 3.2.5), we removed any digits, ‘CC’ (coordinating conjunction), ‘DT’ (determiner), ‘TO’ (TO-infinitive), and ‘IN’ (preposition), and two-letter words from the titles, and labeled the paintings with the remaining 2,048 most frequent words.

All training images have at least one label. If one does not have a label, it is not used as a training sample. Many painting titles can provide informative resources to answer basic questions about the subject matter, such as what, where, who, or when (Gombrich 1985), but during some periods not all titles correlate to content in a helpful way. For example, several titles of Paul Klee (1879-1940) and Joan Miró (1893-1983)’s works refer to literary works, and many other titles in modern art are simply descriptive of shapes or colors, composed of numbers, or images are left untitled. For these reasons, in this project we use the paintings of Renaissance, Baroque, Rococo, Romanticism, Impressionism, Post-Impressionism, and Realism styles.

Fine-Tuning Inception-V3

After modifying the objective of an official model, Inception-V3 (TF-slim in Tensorflow Ver 1.4), we fine-tuned it for 300,000 steps. We only updated the last FC layer, ‘Logits’ and ‘AuxLogits’ (Szegedy et al. 2016), and other parameters were transferred from a pre-trained model. The average loss E' defined in 2 converged from an initial value of $-\log_e(\frac{1}{2048}) = 0.00048$ to a value of about

Table 2: Common Words

PCA region	common words
First and Positive	landscape, river, bridge, path, trees, forest
First and Negative	portrait, child, virgin, Madonna, man, Christ, self, young
Second and Positive	portrait, man, woman, self, young, artist, lady
Second and Negative	landscape, life, trees, beach, scene, winter, bridge

$-\log_e(0.0025)$, but it did not get lower.

The main cause of our high converged error rate is the intrinsic property of titles in artworks. Basically, titles can have various words choices, and even in subject similar paintings, depending on author’s focal points, we can choose words that are semantically different. In other words, there is not only one correct title for an image. Hence, our simple probabilistic output modeling, $P(V|I)$, conditioning only on an input image, may not be sufficient to capture the variance of titles.

Evaluations

We validated our three claims by using the three methodologies described in the Validation Methods section of the Methodology. In the following three subsections, we present the results of the evaluations.

Content Detector: (1) comparison between machine pmfs and words populations

We compared two statistics. The first statistic is the machine output pmf as inputting a simulated image embedding. Four simulated embeddings were computed by conducting inverse PCA approximations on the four vectors: $[4, 0]$, $[-4, 0]$, $[0, 4]$, $[0, -4]$ and from them we gained four pmfs. The pmfs’ 15 top ranked words are presented in four left-handed figures (blue) in Figure 3.

The second statistic is a relative frequency of each title word in a group of images. Four groups of images are collected from the ranges defined in Figure 2. The top ranked 15 words of each group are presented in four right-handed figures (red) in Figure 3.

To consider their similarity, in each row, we compared the left and right figures. Then, we listed the common words in Table 2. We observed that at least six words were common and were semantically aligned with one another, even when they were not perfectly matched. It is natural for them not to be exactly the same each other because the results of the first column are approximately simulated from the first two PCA components, and do not consider all dimensions. Interestingly in Figure 3, there were two considerable concepts: landscape (1st and 4th row) and portrait (2nd and 3rd row). One possible explanation for the result may be the dominant majority of the portrait and landscapes in our data set. In the WikiArt data set, there were 18 different genres, but 37% of the samples were the two genres.

Content Detector: (2) survey results

We conducted a survey to evaluate how the machine’s highly probable words were relevant to an input image. In the survey, we randomly selected 40 validation images and annotated them with the 10 most probable words based on the machine output pmf. It was a blind test and required subjects to do the following (to quote): “Please check all the words that can describe each

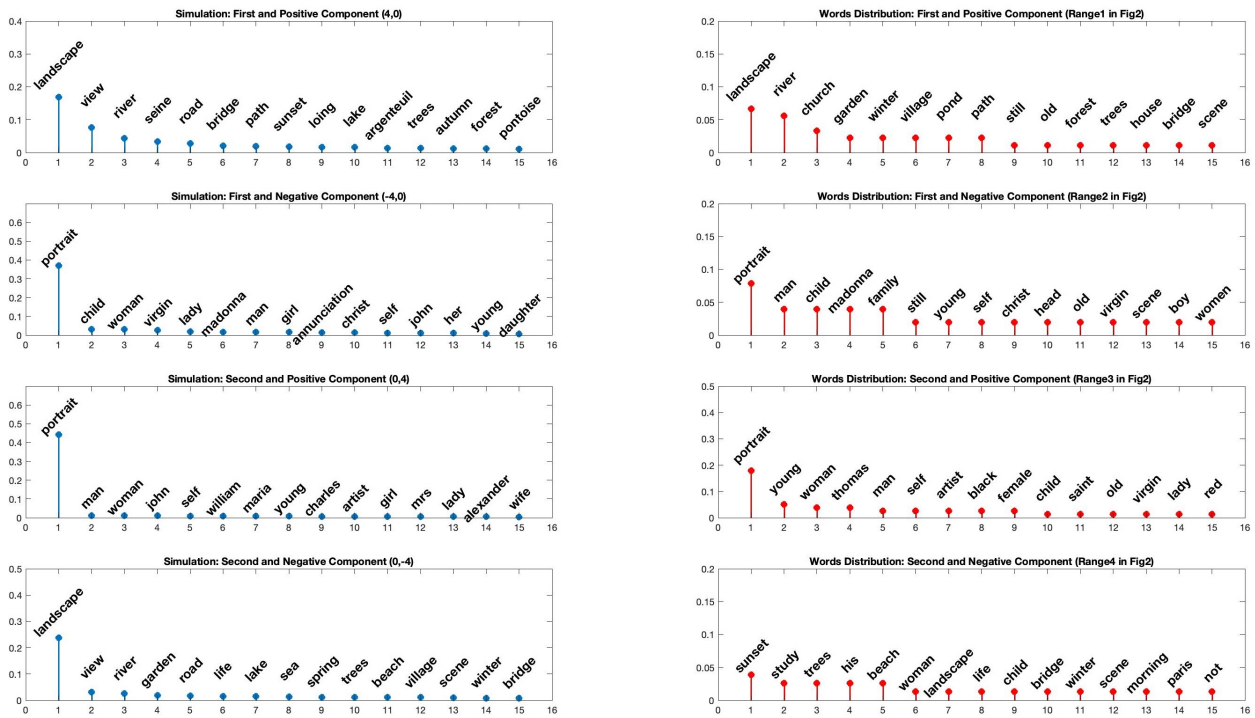


Figure 3: Comparison of two statistics: word detection and word populations within four pre-defined PCA regions

Table 3: Ten Survey Results

Title	Image	P	Machine Annotation (Precision Q = 3 Y = 5)
balchik		1.0	rock(0.8), sea(0.9), cliff(0.9)
the conversation		1.0	portrait(0.5), woman(0.9), girl(0.6)
portrait of old woman		1.0	portrait(0.9), head(0.8), woman(0.9)
country boy		0.67	portrait(0.85), seated(0.85), death(0.15)
annunciation		0.67	virgin(0.9), annunciation(0.85), saint(0.46)

Title	Image	P	Machine Annotation (Precision Q = 3 Y = 5)
a portrait of a christian de falbe		0.33	dog(0.9), woman(0.0), dancer(0.1)
venice		0.33	paix(0.0), house(0.9), bridge(0.1)
cristo no horto		0.0	portrait(0.0), child(0.1), virgin(0.4)
the decline of the Carthaginian empire		0.0	night(0.1), interior(0.0), tavern(0.0)
allegory of air		0.0	jerome(0.1), portrait(0.0), dancing(0.0)

painting. Do not check any words if none are relevant.” At least 12 graduate students in art history responded for each of the survey images.

We set thresholds from 0 to 1 with a step size of 0.1, and obtained a correct word set based on the levels. For example, for a threshold of 0.3, we considered words as right answers only when more than three out of 10 people agreed with

the word. Let the Q denote the number of top words and $q_u(Y)$ the number of correct words at threshold Y . Then, a precision@ Q at threshold Y over the $U = 40$ images can be defined as

$$P@Q(Y) = \frac{1}{U} \sum_u \frac{q_u(Y)}{Q} \quad (6)$$

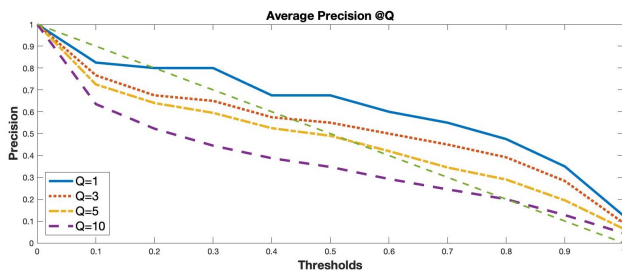




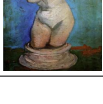


Figure 4: Survey Average Precision Rates $P@Q = 1, 3, 5,$ and 10

Table 4: Author Selected Test Result

Title	Image	Machine 10 most probable words
the bouquet		portrait, rose, woman, girl, lady flowers, young, red, roses, miss
garden in bloom		apple, trees, oak, orchard, blossom park, bloom, tree, landscape, grove
villa torlonia fountain		view, bridge, evening, park, landscape garden, fountain, pond, street, gardens
mary with child		child, madonna, portrait, girl, virgin woman, young, lady, peasant, maria
plaster statuette of a female torso		torso, blue, still, woman, life, portrait, jug, study, nude, plaster

In Figure 4, we present the precision results for $Q=1, 3, 5,$ and 10. As Q increases, the precision values decrease and when the threshold is 0.5, the precision values are 0.68, 0.55, 0.5, and 0.35 at $Q = 1, 3, 5,$ and 10. This result validates the performance of our content detector in two senses. First, the most probable word shows a 68% average precision rate as we set the right words only when more than half of subjects agree on them. Second, as the Q increases, the corresponding precision rate drops. It implies that the machine's less probable words do not contribute to increasing the precision rate. Hence, we can see the ranking of words in the machine pmf is correlated with the subjects' responses.

To examine the quality of our system, we listed ten survey results ($Q = 3$ and $Y = 0.5$) in order of the precision rates in Table 3 and characterized them. For the high-rated (left-hand) results, most are expressed typically and simply in terms of each genre. On the other hand, for low-rate (right-hand) examples, their main figures are expressed as relatively small in complex circumstances, or a portion of the figures has characteristics that often represent other content. For instance, the third 'cristo no horto' depicts Christ,

Table 5: Number of GoogleArt&Culture Search Results

Group	Group 1	Group 2	Group 3
Averaged number of results	12,805	4,983	3,554

Table 6: Descending ordered words

Word	15 relevant words
tree	trees, pine, oak, olive, bloom, pines, orchard landscape, oaks, blossom, grove, willow, forest, asylum, peach
christ	cross, lamentation, angels, homo, ecce, deposition, virgin holy, adoration, saints, baptist, entombment, ancestors, jesus, crucifixion
angel	virgin, annunciation, vision, baptist, angels, penitent, tobias madonna, resurrection, magdalene, jesus, death, creation, elijah, allegory
rose	bouquet, wildflowers, flowers, roses, lilies, pink hollyhocks, violets, irises, lily, vase, Japanese, nasturtiums, iris, daisies
nude	female, seated, reclining, standing, bather, bath, naked model, back, hair, woman, torso, nudes, herself, male
lighthouse	seascape, tide, sunset, sailboats, lunar, harbor tower, marseille, moonrise, calm, coast, channel, maggiore, steppe, newport
virgin	madonna, child, assumption, holy, coronation, saints angels, annunciation, adoration, christ, mary, trinity, birth, enthroned, baptist

but he wears a mantle of blue, which often represents his mother. He may be wrongly detected as the virgin Mary. In the second example, 'venice', the rail of the window may be the reason why the machine detects the bridge as the third word. For further references, we selected five test-set results in Table 4. For each example, the 10 most probable words were annotated based on the machine's pmf outcome.

Words Distributed Representation As described in the Validation Methods in Methodology, by pairing two words (a word of group 0 and another word from one of the groups 1, 2, or 3), we searched GoogleArt&Culture and averaged the number of returned art works for each group. In the experiment, we only considered the top 400 words among the machine's 2,048 output domain words. The upper words are more frequent and account for more than 65% of the words frequencies, so we regard them as a representative set. The three groups' results were averaged over 400 words and presented in Table 5. It shows that the number of results decreases by 60% from group 1 to 2 and by 28% from group 2 to 3. Hence, based on the assumption that having more search results implies more connections, we can argue that closely distributed representations are likely to represent stronger relationships between words. We presented seven examples in Table 6. Based on distance analysis, we enumerated the 15 closest words for each example.

Discussion

Nowadays, many museum websites provide services to allow web users to search their digital collections through matching the user's words to basic text information they already have. The text description can refer to the title, author, genre, time period, style, or sometimes detailed documentation written by curators or art historians, but there are limited ways to search for images beyond the given categories. To do so, the user must already know what they are looking for and deploy the correct keywords, both of which require highly specialized knowledge.

However, if we can search the images aligned with their content, then all users will be able to access the database,

and search using a broader and more comprehensive scope. For example, a user could search for all 19th-century French landscape paintings, either winter scenes and summer scenes, with or without figures, etc., and locate all the works in the large database without failing to locate relevant images.

Distributed representation can also be useful to suggest other relevant concepts to user's search words. For example, when we look for a specific book, browsing nearby shelves can sometimes produce a more useful book even if the book is titled with words that we do not initially consider. In art searches, we cannot access the physical storage of the works, but instead we gain information about links between content words, thereby connecting a larger number of art works to our search.

Conclusion

In this paper, we introduced the first deep-learning approach to computationally analyze the contents in fine art paintings. Motivated by significant performances and broad adaptability of deep neural networks in computer vision, we adopted the Inception-V3 as the primary model of our content detector, validated its performance, and considered its last layer parameters as informative resources related to content. In general, the system showed positive correlations with survey responses, but limitations regarding certain types of paintings especially in complex depictions or compositions. To refine our models, we are still looking at other advanced deep-learning algorithms. For example, beyond words, we could build a system to describe art using natural language. A recurrent neural network on top of our system would be a feasible example (Vinyals et al. 2015). Furthermore, the current system perceives the whole image at once, but as content in paintings is often spatially local rather than global, principles in scene labeling (Farabet et al. 2013) or attention modeling (Xu et al. 2015) are expected to provide more sophisticated boards for computational content analysis.

References

Bown, O. 2012. Generative and adaptive creativity: A unified approach to creativity in nature, humans and machines. In *Computers and creativity*. Springer. 361–381.

Carneiro, G.; da Silva, N. P.; Del Bue, A.; and Costeira, J. P. 2012. Artistic image classification: An analysis on the printart database. In *European Conference on Computer Vision*, 143–157. Springer.

Carneiro, G. 2011. Graph-based methods for the automatic annotation and retrieval of art prints. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, 32. ACM.

Crowley, E. J., and Zisserman, A. 2014. In search of art. In *Workshop at the European Conference on Computer Vision*, 54–70. Springer.

Dyke, J. C. V. 1887. *Principles of Art*. New York: New York, Fords, Howard, Hulbert.

Elgammal, A.; Liu, B.; Kim, D.; Elhoseiny, M.; and Mazzone, M. 2018. The shape of art history in the eyes of the

machine. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Elgammal, A.; Kang, Y.; and Den Leeuw, M. 2018. Picasso, matisse, or a fake? automated analysis of drawings at the stroke level for attribution and authentication. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Farabet, C.; Couprie, C.; Najman, L.; and LeCun, Y. 2013. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence* 35(8):1915–1929.

Gombrich, E. H. 1985. Image and word in twentieth-century art. *Word & image* 1(3):213–241.

Hendriks, E., and Hughes, S. 2009. *Van Goghs brushstrokes: Marks of authenticity?*

Jafarpour, S.; Polatkan, G.; Brevdo, E.; Hughes, S.; Brasoveanu, A.; and Daubechies, I. 2009. Stylistic analysis of paintings using wavelets and machine learning. In *Signal Processing Conference, 2009 17th European*, 1220–1224. IEEE.

Kim, D.; Liu, B.; Elgammal, A.; and Mazzone, M. 2018. Finding principal semantics of style in art. *IEEE International conference on semantic computing*.

Lowry, B. 1967. *The visual experience : An introduction to Art*. Englewood Cliffs New Jersey: Prentice-Hall, INC. and HARRY N. ABRAMS, INC.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Munsterberg, M. 2009. *Writing about Art*. <http://writingaboutart.org>: New York, Fords, Howard, Hulbert.

Panofsky, E. 1971. *Early Netherlandish painting: its origins and character*, volume 1. Natl Gallery of Art.

Picard, D.; Gosselin, P.-H.; and Gaspard, M.-C. 2015. Challenges in content-based image indexing of cultural heritage collections. *IEEE Signal Processing Magazine* 32(4):95–102.

Polatkan, G.; Jafarpour, S.; Brasoveanu, A.; Hughes, S.; and Daubechies, I. 2009. Detection of forgery in paintings using supervised learning. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, 2921–2924. IEEE.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–2826.

Vinyals, O.; Toshev, A.; Bengio, S.; and Erhan, D. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3156–3164.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, 2048–2057.

Learning to Surprise: A Composer-Audience Architecture

Razvan C. Bunescu and Oseremen O. Uduehi

School of Electrical Engineering and Computer Science

Ohio University

Athens, OH 45701

bunescu,ou380517@ohio.edu

Abstract

The ability to generate surprising outputs is essential for creative behavior. Surprise, or violation of expectation, has been hypothesized to be part of a fundamental mechanism enabling the capacity for emotion found in creative fields such as music, art, humor, or literature. Machine learning approaches to music generation train one model and sample from its distribution to generate new outputs. We show that this one-model sampling is fundamentally limited in its capacity for surprise. Drawing on insights from music and humor understanding, we propose a two-model architecture composed of an audience model for learning expectations connected to a composer model for learning to surprise. The new architecture facilitates a natural measure for surprise that is used in experimental evaluations on a set of synthetic tasks with binary strings. When instantiated with neural networks, the composer-audience model is shown to successfully learn to generate deterministic or random patterns of surprise, demonstrating its potential as a general framework for machine learning approaches to creative processes.

Introduction and Motivation

Creativity is widely considered to be an essential component of intelligent behavior (Boden 1991; Grace and Maher 2015). Surprise is a powerful driver for creativity and discovery, as such it has been used to guide search algorithms in models of computational creativity and discovery (Yanakis and Liapis 2016). Owing to its importance for the creative process, surprise has become one of the core criteria for the evaluation of creative artifacts, together with novelty and value (Grace et al. 2015). As reviewed in (Itti and Baldi 2009), surprise is an essential concept in many studies on the neural basis of behavior, with surprising stimuli shown to be strong attractors of attention.

Surprise, or violation of expectation, has also been hypothesized to be an essential mechanism through which music and stories elicit emotion. According to (Meyer 1961), the principal emotional content of music arises from the composers manipulation of expectation. Composers build expectations in time, which then they purposely violate in order to elicit tension, prediction, reaction, and appraisal responses (Huron 2008). While significant progress has been made towards models that learn harmony, voice leading, and

even long-term structure, e.g. (Boulanger-Lewandowski, Bengio, and Vincent 2012), (Hadjeres, Pachet, and Nielsen 2017), (Oore et al. 2018), (Huang et al. 2019), the importance of surprise for eliciting emotion is not reflected in the design of machine learning (ML) approaches to music generation, which use sampling from the trained model distribution to generate new musical output. In this paper, we show that one-model sampling is fundamentally limited in its capacity to generate surprising outputs and propose an architecture for learning to surprise comprised of an audience model that learns expectations and a composer model that learns patterns of violations of expectations. We instantiate the proposed architecture with LSTMs and evaluate it on three synthetic tasks with binary strings. Experimental results show that sampling from the two-model architecture enables much higher levels of surprise when compared with traditional one-model sampling. We conclude the paper by positioning our model in the context of previous work.

One-Model Sampling is Unsurprising

The vast majority of ML models used for generative tasks train **one model** P_M on **one dataset** D that is sufficiently large to enable a good approximation of the true data distribution. Given a trained model P_M , a sampling procedure is used to generate an output $\hat{\mathbf{x}} \sim P_M(\mathbf{x})$. When a language modeling approach is used for sequences of discrete events $\mathbf{x} = \langle x_1, x_2, \dots \rangle$, such as tokens in text generation or chords in music composition, it is common to use a left-to-right sequential sampling based on the factorization below:

$$P_M(\mathbf{x}) = \prod_{k=1}^{|\mathbf{x}|} P_M(x_k | \mathbf{h}_k) = \prod_{k=1}^{|\mathbf{x}|} P_M(x_k | x_{k-1}, \dots, x_1)$$

Such factorization could be provided for example by a unidirectional RNN. To generate an output sequence from left to right, at every step k a token \hat{x}_k is sampled according to the model, i.e. $\hat{x}_k \sim P_M(x_k | \mathbf{h}_k)$, where $\mathbf{h}_k = \langle \hat{x}_{k-1}, \dots, \hat{x}_1 \rangle$ is the history (or context) of previously sampled tokens. If x_k itself is high-dimensional, as is the case with chords in music, models such as the restricted Boltzman machine (RBM) (Smolensky 1986; Hinton 2002) or the neural autoregressive distribution estimators (NADE) (Larochelle and Murray 2011) can be used to compute $P_M(x_k | \mathbf{h}_k)$ and generate approximate or exact samples, respectively. More complex

factorizations of the distribution, such as the bi-directional model of DeepBach (Hadjeres, Pachet, and Nielsen 2017) or general probabilistic graphical models require more sophisticated sampling procedures, e.g. MCMC methods, variational methods, or sampling via random projections. DeepBach, for example, uses a pseudo-Gibbs sampling procedure where at every iteration a note k is selected at random and then its pitch value $\hat{x}_k \sim P_M(x_k|\mathbf{h}_k)$ is re-sampled. In this case the history \mathbf{h}_k contains all the other notes in the piece.

It is possible for this type of **one-model sampling** to generate surprising events. For example, if x_k is a binary variable and $P_M(x_k = 1|\mathbf{h}_k) = 0.9$, then on average 1 out of 10 times the sampling procedure will generate the "surprising" event $\hat{x}_k = 0$. The following informal observations can be made from this example:

Remark 1. Generation of *surprise* is possible only in event spaces with non-uniform probability distribution.

Remark 2. The *more surprising* an event needs to be, the less likely it is for it to be generated by one-model sampling.

Based on the later, one-model sampling would be ill-suited for tasks that require generating surprising events with high probability. Furthermore, one-model sampling generates surprising events in a completely random manner, which can be deeply unsatisfying if the task requires control over when to generate surprise or learning patterns of surprise.

We now present more formal versions of these statements, together with the corresponding proofs. While the two informal remarks seem obviously true, a formal specification has the advantage that it quantifies the notion of *surprise*, which will also help in clarifying the meaning of *more surprising* and its connection to sampling.

Definition 1. Let $\psi \in (0, 1]$ be an *expectation level*, with lower levels used to represent higher surprise. Let M be a model that computes the categorical distribution $P_M(x|\mathbf{h})$ over K categories. A discrete event x observed in a context \mathbf{h} is called ψ -surprising for model M if $P_M(x|\mathbf{h}) < \psi/K$.

Definition 2. Let $S(\psi, \mathbf{h}, M)$ denote the expectation under M that a sampled event $\hat{x} \sim P_M(x|\mathbf{h})$ is ψ -surprising for M :

$$S(\psi, \mathbf{h}, M) = \mathbb{E}_{\hat{x} \sim P_M} [\mathbb{I}[P_M(\hat{x}|\mathbf{h}) < \psi/K]]$$

where we use the Iverson bracket $\mathbb{I}[P] = 1$ if the proposition P is satisfied, and 0 otherwise.

Using these definitions, the formal versions of the two remarks above are expressed as Theorems 1 and 2 below.

Theorem 1. $S(\psi, \mathbf{h}, M) = 0$ for any uniform categorical distribution M , irrespective of the level $\psi \in (0, 1]$.

Proof. If M is a uniform categorical distribution, $P_M(\hat{x}|\mathbf{h}) = 1/K$. Therefore $\mathbb{I}[P_M(\hat{x}|\mathbf{h}) < \psi/K] = 0$ for any $\hat{x} \sim P_M(x|\mathbf{h})$, which means that the corresponding ψ -surprising expectation is $S(\psi, \mathbf{h}, M) = 0$. \square

According to Theorem 1, a maximum entropy distribution offers no opportunity for surprise.

Theorem 2. Let M be a non-uniform categorical distribution and $\rho_k = P_M(x = k|\mathbf{h})$ for each category $k \in 1..K$. Without loss of generality assume that the categories are

indexed in the order of their sampling probabilities, i.e. $\rho_0 = 0 \leq \rho_1 \leq \rho_2 \leq \dots \leq \rho_K \leq 1$. Then:

(a) $\frac{\psi}{K} \in (\rho_k, \rho_{k+1}]$ for some $k \in 0..K-1$.

(b) $S(\psi, \mathbf{h}, M) = \sum_{j=0}^k \rho_j \leq \psi \frac{k}{K}$.

Proof. To prove (a) note that $\psi \in (0, 1] \wedge K \geq 2 \Rightarrow \psi/K \in (0, 1/K] \subset (0, 1]$. This means that ψ/K must belong to one of the sub-intervals from the following partition of $(0, 1]$:

$$(0, 1] = (0, \rho_1] \cup \bigcup_{k=1}^{K-1} (\rho_k, \rho_{k+1}] \cup (\rho_K, 1]$$

However, it is not possible for ψ/K to belong to the last sub-interval. Since M is non-uniform and ρ_K is the largest, this means $\rho_K > 1/K \geq \psi/K \Rightarrow \psi/K \notin (\rho_K, 1]$. Therefore, there can be only two scenarios:

1. $\psi/K \in (0, \rho_1]$, if $k = 0$ in (a).
2. $\psi/K \in (\rho_k, \rho_{k+1}]$, for some $k \in 1..K-1$ in (a).

We prove (b) separately for each of the two cases. In scenario 1, $\psi/K \leq \rho_1$ implies $\psi/K \leq \rho_k$ for all categories $k \in 1..K$. Therefore, $\mathbb{I}[P_M(\hat{x}|\mathbf{h}) < \psi/K] = 0$ for any \hat{x} , which means that the expectation of a ψ -surprising event is:

$$S(\psi, \mathbf{h}, M) = 0 \quad (1)$$

Since $k = 0$ in this scenario, this means $S(\psi, \mathbf{h}, M) = \psi k/K$ which satisfies (b).

In scenario 2, because of how the categories were indexed, we have $\rho_1 \leq \rho_2 \leq \dots \leq \rho_k < \psi/K \leq \rho_{k+1}$. Thus, for $P_M(\hat{x}|\mathbf{h}) < \psi/K$ to be true, \hat{x} must satisfy $1 \leq \hat{x} \leq k$. Therefore, the ψ -surprising expectation is:

$$\begin{aligned} S(\psi, \mathbf{h}, M) &= \mathbb{E}_{\hat{x} \sim P_M} [\mathbb{I}[P_M(\hat{x}|\mathbf{h}) < \psi/K]] \\ &= \sum_{j=1}^k P_M(\hat{x} = j|\mathbf{h}) \\ &= \sum_{j=1}^k \rho_j \leq \sum_{j=1}^k \rho_k = k\rho_k < \psi \frac{k}{K} \end{aligned} \quad (2)$$

which satisfies (b). \square

Corollary 2.1 below expresses the fact that generating very surprising events (small ψ/K) is impossible in the absence of very unlikely events (smaller $\rho_1 < \psi/K$).

Corollary 2.1. $S(\psi, \mathbf{h}, M) = 0$ if $\psi/K \leq \rho_1$.

The dependence on the categorical distribution M can be removed from Theorem 2, as shown in Corollary 2.2 below.

Corollary 2.2. $S(\psi, \mathbf{h}, M) \leq \psi \left(1 - \frac{1}{K}\right) < \psi$.

For Bernoulli distributions $K = 2$, for which the bound $S(\psi, \mathbf{h}, M) \leq \psi/2$ in the theorem is tight.

Overall, Theorem 2 and its corollaries show that it is impossible for one-model sampling to generate very surprising events (i.e. very low level ψ) with high probability (i.e. large expectation $S(\psi, \mathbf{h}, M)$).

Why Controlled Surprise is Important

Using one-model sampling as the sole means of generating surprise has therefore two fundamental limitations:

1. Generation of truly surprising events, i.e. ψ -surprising with very small ψ , is very unlikely.
2. Surprising events are generated completely at random, with no mechanism available to (learn to) control surprise generation.

To better understand why the two limitations are important, consider the task of generating satirical news headlines. As observed by West and Horvitz (2019), changing a single word in a satirical news headline is often sufficient to make it sound like serious news, as in "BP ready to resume oil {spilling, drilling}". Furthermore, the changed word tends to reside towards the end of the headline. Using the notation introduced earlier, the context \mathbf{h} can be seen as carefully building an expectation in the audience that is then turned upside-down by the word x_k appearing at position k towards the end of the headline. If M is the reader's model of expectation and V is the vocabulary, this can be expressed as $P_M(x_k|\mathbf{h}) < \psi/|V| \ll 1/|V|$, i.e. x_k is ψ -surprising for M with ψ very small. However, according to Theorem 2, a very small ψ makes it highly unlikely that sampling from a trained audience model would generate such a surprising event. Even when surprise is allowed at any position in the headline, the overall likelihood of sampling a surprising word is still very small because satirical headlines are usually very short. Thus, a writer generating headlines by sampling from the language model would have to discard a very large number of outputs before stumbling upon a satirical one. However, this is not how writers generate satirical headlines: while some randomness is probably still part of the process, there is also a significant mechanism at play that makes the generation of surprise substantially more likely than mere sampling from a language model shared with the audience.

The second limitation can manifest in multiple ways, for example as an inability to determine the required level of expectation violation or the frequency of surprising events. To illustrate, consider a headline generator primed with the word "BP" that generates phrases sequentially as shown below:

BP \Rightarrow BP wind farms
 \Rightarrow BP wind farms to provide
 \Rightarrow BP wind farms to provide grazing land
 \Rightarrow BP wind farms to provide grazing land to nearby ranchers for free.

In the first step, it samples "wind farms", which happens to be just a bit surprising, as it is less expected to appear after "BP" than other phrases, such as "oil tankers". At the next step the high expectation verb "to provide" is sampled. Then the model samples "grazing land" which too happens to be just a bit more surprising in this context than other phrases, such as "electricity". Finally, the model samples relatively high expectation phrases, resulting in the complete headline "BP wind farms to provide grazing land to nearby ranchers for free". The level of surprise in this headline is much

milder than in the satirical "BP ready to resume oil spilling", where "spilling" is much less expected than "drilling" given the previously generated words. While this type of control over the level of surprise is not available in one-model sampling, it can be achieved using a two-model architecture, as described in the next section.

A Two-Model Architecture for Surprise

To enable a data-driven control over the generation of surprise, we propose an architecture that contains two models: an **audience model** M^a and a **composer model** M^c that has access to expectations computed by M^a . These models will be trained on separate datasets, D^a and D^c , respectively. While the definition of ψ -surprising events remains the same as in the one-model sampling case, the definition of the ψ -surprising expectation is generalized to accommodate the two models, as shown in Definition 3 below.

Definition 3. Let $S(\psi, \mathbf{h}, M^a|M^c)$ denote the expectation that an event $\hat{x} \sim P_M^c(x|\mathbf{h})$ sampled from a composer model M^c is ψ -surprising for an audience model M^a :

$$S(\psi, \mathbf{h}, M^a|M^c) = E_{\hat{x} \sim P_M^c} [\mathbb{I}[P_M^a(\hat{x}|\mathbf{h}) < \psi/K]]$$

where the Iverson bracket $\mathbb{I}[P]$ = 1 if the proposition P is satisfied, and 0 otherwise.

The previous Definition 2 can be obtained from 3 by using the audience model also as a composer model, i.e. $M^c = M^a = M$. The adaptation of Theorem 1 for the two-model case still holds:

Theorem 3. $S(\psi, \mathbf{h}, M^a|M^c) = 0$ for any uniform categorical distribution M^a , irrespective of $\psi \in (0, 1]$.

The corresponding version of Theorem 2 (b) however does not hold anymore, as Equation 2 now changes to Equation 4:

$$S(\psi, \mathbf{h}, M^a|M^c) = \sum_{j=1}^k P_M^c(\hat{x} = j|\mathbf{h}) \quad (4)$$

While $S(\psi, \mathbf{h}, M^a|M^c)$ is still 0 for $\psi/K \leq \rho_1 = \min_k P_M^a(\hat{x} = k|\mathbf{h})$, it can now become arbitrarily large when $\psi/K > \rho_1$, depending on how much probability the composer allocates to categories that are unlikely in the audience distribution.

In order to learn to control violations of expectations (VoE), the composer model uses as input expectations computed by the audience model. We use the notation $M^c \leftarrow M^a$ to show this dependency. It is important that the two models are trained separately, on different datasets, in this sequence: M^a is first trained on data D^a , then it is plugged in the $M^c \leftarrow M^a$ architecture for training M^c on its dataset D^c while keeping M^a fixed. This training procedure is shown in Algorithm 1. During training of the M^c model in the $M^c \leftarrow M^a$ architecture, the same composer sequence x_k^c is provided as input to both M^a and M^c . To generate samples from the composer at test time, the previously trained M^a can be used or a new one can be trained on a different audience dataset D , as shown in Algorithm 2. When used in generation mode at test time, a token \hat{x}_t^c is sampled according to the categorical distribution P_t^c and

Algorithm 1 TRAINCAMODEL(D^a, D^c)

Input: Datasets D^a and D^c .**Output:** Composer model M^c .

- 1: train M^a on D^a
 - 2: train $M^c \leftarrow M^a$ on D^c ▷ Keep M^a fixed.
 - 3: **return** M^c
-

Algorithm 2 TESTCAMODEL(M^c, D)

Input: Composer model M^c .**Input:** New audience examples D .**Output:** Samples \hat{x}^c from composer (and \hat{x}^a from audience).

- 1: train M^a on D
 - 2: sample $\hat{x}^c \sim M^c \leftarrow M^a$ (and $\hat{x}^a \sim M^a$)
 - 3: **return** \hat{x}^c (and \hat{x}^a)
-

fed as the next input token to both M^a and M^c . We use $\hat{x}^c \sim M^c \leftarrow M^a$ to denote the entire sequence sampled from the composer, whereas $\hat{x}^a \sim M^a$ is used to refer to a sequence sampled from the audience model.

The composer-audience (CA) architecture $M^c \leftarrow M^a$ can be instantiated using various ML models, depending on the type of data that needs to be processed. Figure 1 shows the architecture used in our experiments with synthetic data, which relies on LSTM units for processing sequential data. At the bottom, the audience LSTM processes the input sequence and computes at each time step $t - 1$ a hidden state h_{t-1} and the categorical distribution P_t^a over the possible values for the next token. We call this the audience expectation for the next token. Together with the current token in the sequence, this expectation is used as input to the composer LSTM shown at the top of the figure. Optionally, the hidden state of the audience model could also be provided as input to the composer. The composer LSTM then computes its own categorical distribution P_t^c over the possible values for the next token.

Possible Scenarios for Text and Music

For the computational humor task described earlier, D^a could be a large collection of news headlines, perhaps augmented with text from news articles or open domain text. When trained on it, the M^a model would capture the expectation $P_M^a(x|\mathbf{h})$ of seeing word or phrase x in a textual context \mathbf{h} in a normal, largely *non-humorous* text. The dataset D^c on the other hand would be composed only of satirical news headlines. Accordingly, training the composer model M^c on D^c using the CA architecture $M^c \leftarrow M^a$ would enable M^c to learn patterns of violations of expectations, such as generating a word that is ψ -surprising for M^a only when the audience expectation $P_M^a(x|\mathbf{h})$ for other words is very large. When used in the $M^c \leftarrow M^a$ architecture, the composer will also be able to learn the tendency for surprising words to be generated towards the end of the headline. In contrast, as shown earlier, training only one model on D^a will have very limited capacity for surprise and offer no control over when to violate expectations. Training one model on D^c is not going to work either, as it will not be able to

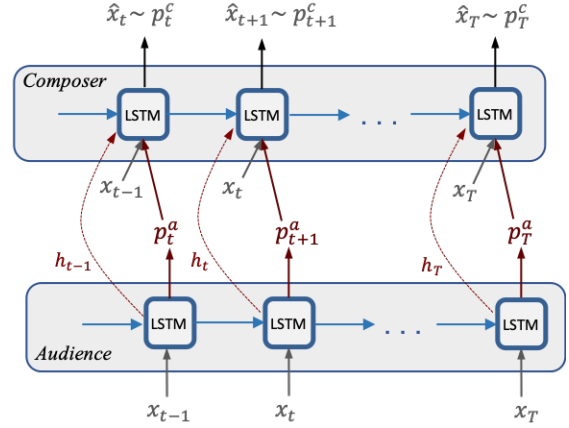


Figure 1: LSTM instantiation of $M^c \leftarrow M^a$ architecture.

distinguish between words that are expected by an audience and words that violate the audience’s expectations.

The two-model architecture could be applied in a similar way to music generation. The common practice is to train one model on one corpus of music D and generate music by performing one-model sampling from it, a method that has limited capacity for surprise, as shown earlier. Instead, we propose training a composer-audience model, for which the music corpus needs to be partitioned into an audience corpus and a composer corpus, i.e. $D = D^a \cup D^c$. One solution is to choose a “present” time t and partition D around that time, i.e. store all music composed before t in the “old music” dataset D^a whereas all music composed after t is stored in the “new music” dataset D^c . After training the $M^c \leftarrow M^a$ architecture on this partition, the audience model would be re-trained on the entire dataset, plugged back in the $M^c \leftarrow M^a$ architecture, and new music would be generated by sampling from M^c in this architecture. The method can be refined in many ways, such as moving into D^a all music from D^c that is deemed too similar or derivative with respect to D^a . Furthermore, the fixed time cutoff for the partition can be avoided, as shown in the approach below aimed at addressing individual differences.

The Personalized Composer While it has the advantage of being simple, training just one $M^c \leftarrow M^a$ model does not consider individual differences in humor appreciation or music enjoyment. With respect to music, significant individual differences exist, from individuals who tend to experience a complex array of intense physiological and mental responses (Panksepp 1995) to individuals who report being unable to derive pleasure from listening to music (Mas-Herrero et al. 2014). While individual differences in aesthetic reward sensitivity were shown to have a neural basis (Sachs et al. 2016), differences in musical ability and familiarity were also observed to be important for experiencing intense emotional responses to music (Nusbaum and Silvia 2011). If we use the individual’s performing or listening history $D = \{x_1, x_2, \dots, x_T\}$ as a proxy for their musical ability and familiarity, then the CA architecture can be

used to train a composer model using a series of audience models. As shown in Algorithm 3 below, at each timestep t an audience model M_t^a is trained on all current music $D_t^a = \{x_1, \dots, x_t\}$ and plugged in the $M^c \leftarrow M_t^a$ architecture to create together with x_{t+1} a training example for the composer model.

Algorithm 3 TRAINCASERIES(D)

Input: Chronological dataset $D = \{x_1, x_2, \dots, x_T\}$.

Output: Composer model M^c .

- 1: **for** $t = 1$ to $T - 1$ **do**
 - 2: train M_t^a on $D_t^a = \{x_1, \dots, x_t\}$
 - 3: train M^c on examples $\{M^c \leftarrow (M_t^a, x_{t+1})\}_{t=1..T-1}$
 - 4: **return** M^c
-

Experimental Evaluation

The proposed CA architecture was evaluated on three synthetic tasks using binary strings: 1) violation of all high expectations, 2) violation followed by resolution of expectation, and 3) self-perpetuating random VoE. These synthetic tasks use clear patterns of expectations that enable us to determine the extent to which the models learn the expectations (audience) and their patterns of violation (composer). We use two evaluation measures for surprise throughout:

1. Expected maximum surprise:

$$S_{max}(M^a|M^c) = E_{\hat{x} \sim P_M^c} \left[1 - \min_{1 \leq j \leq |\hat{x}|} P_M^a(\hat{x}_j|\mathbf{h}) \right]$$

2. Expected count of ψ -surprise:

$$S_{cnt}(\psi, M^a|M^c) = E_{\hat{x} \sim P_M^c} \left[\left| \left\{ \hat{x}_j \mid P_M^a(\hat{x}_j|\mathbf{h}) < \frac{\psi}{K} \right\} \right| \right]$$

S_{max} and S_{cnt} are calculated by averaging the sequence-level maximum surprise or count, respectively, over a set of generated sequences. Because all generated strings \hat{x} in the experiment have the same fixed length N , the averaged count $S_{cnt}(\psi, M^a|M^c)/N$ can be seen as an estimate of the average ψ -surprising expectation from Definition 3.

The two models are trained using *teacher forcing*, i.e. the true token x_t is used as input for the next step. We use the cross entropy loss with respect to all the bits (random or pattern) in the training sequences. We emphasize that there is no explicit surprise-related loss and the only means for the composer to learn surprise is from the data. The extent to which the trained composer surprises the audience reflects the extent to which the patterns in D^c violate the expectations of a model trained on the patterns in D^a .

Violation of Expectation

In this scenario, the audience model learns when to generate high expectations, whereas the composer model learns to violate all expectations that are sufficiently high, where the expectation level required for VoE is learned from the data. Training and test examples are generated as quasi-random sequence of bits that are constrained to contain a

	Training patterns		Test
Audience	$x_1^a(0011)$	$x_2^a(1100)$	$x^a(0101)$
Composer	$x_1^c(0010)$	$x_2^c(1101)$	$x^c(0100)$

Table 1: Audience & Composer examples for VoE.

Algorithm 4 PAIREDTRAINING(D^a, D^c)

Input: Audience dataset $D^a = \{x_1^a, \dots, x_K^a\}$.

Input: Composer dataset $D^c = \{x_1^c, \dots, x_K^c\}$.

Output: Composer model M^c .

- 1: **for** $k = 1$ to K **do**
 - 2: train M_k^a on $\{x_k^a\}$
 - 3: train M^c on examples $\{M^c \leftarrow (M_k^a, x_k^c)\}_{k=1..K}$
 - 4: **return** M^c
-

given bit pattern. To compress the dataset description, let $x_j^m(b_1 b_2 \dots b_k b_{k+1}, p, N)$ specify that example number j for the audience ($m = a$) or the composer ($m = c$) is a sequence of N bits that are generated at random with the following constraints:

1. Any time the sequence of bits $\langle b_1, b_2, \dots, b_k \rangle$ is generated, it is immediately followed by the bit value b_{k+1} .
2. The complete pattern $\langle b_1, b_2, \dots, b_k, b_{k+1} \rangle$ appears pN times in the entire string of N bits ($p < 1$).

For the rest of the paper we will be dropping the pattern frequency p and length N from the notation, as these will be global parameters that stay the same for all examples: $p = 0.1$ and $N = 200$. Table 1 shows training patterns used for the audience and composer in the experiments for this section. Below we show example training sequences generated for two patterns, one for the audience and one for the composer:

$$\begin{aligned} x_1^a(0011) &= \langle 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, \dots \rangle \\ x_1^c(0010) &= \langle 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, \dots \rangle \end{aligned}$$

The composer patterns are designed to be the "opposite" of the audience patterns: any antecedent string 0, 0, 1 in the audience sequence above is followed by the consequent bit 1. For the composer sequence, the antecedent is the same but the consequent is flipped to 0. By training on this data, the composer is expected to learn that whenever the audience expectation is high (i.e. for a consequent bit) it should go against that expectation and generate the opposite bit.

We experimented with two training scenarios:

1. **Paired training:** In this scenario, shown in Algorithm 4, a separate audience model is trained for each pattern. When training the composer model on a composer pattern x_k^c , we plug in the audience model that was trained on the corresponding pattern x_k^a .
2. **Unpaired training:** This is the original training scenario shown in Algorithm 1 which is more realistic, as it does not require having knowledge of which patterns are used during training.

For paired training, we used the $K = 2$ training patterns shown in Table 1. For the more difficult case of unpaired

training we added 3 more patterns to enable the composer model to better learn the importance of audience expectation: $x_3^a(1101)$, $x_4^a(1010)$, $x_1^a(0110)$ for the audience, and the corresponding opposite patterns for the composer. At test time, the audience model is trained on the new pattern $D = \{x^a\}$ and plugged in the $M^c \leftarrow M^a$ architecture, as shown in Algorithm 2. When used for sampling, the M^c model is expected to generate a string \hat{x}^c that violates the expectation engendered by this new pattern. Note that the M^c model does not see the new pattern x^a during training.

The LSTMs were trained with Adam (Kingma and Ba 2015) for 10,000 epochs using a learning rate of 0.001. We generated 1,000 of training sequences of 200 bits each, for each pattern. The LSTM had two layers of neurons, with 2 neurons per layer for both the audience and composer model in the paired training mode. For unpaired training, where just one model had to learn all patterns, these were increased to 5 neurons for the audience and 4 for the composer. Overall, it was important to keep the capacity small so that the audience does not memorize the input sequences, while ensuring that the composer does not overfit to the bit pattern. We report results for two input scenarios for the composer:

1. Using only the audience expectation as input (first result in the table cells).
2. Using both the bit at the current position and the audience expectation (results in parentheses in the table cells).

Sampling model M	$S_{max}(M^a M)$	$S_{cnt}(\psi, M^a M)$
Audience: $M = M^a$	0.61 (0.64)	0.07 (0.13)
Composer: $M = M^c$	0.99 (0.99)	11.03 (12.14)

Table 2: Expected ψ -surprise for Paired training: $\psi = 0.1$, S_{cnt} is per 100 bits, composer accuracy 99% (99%).

Sampling model M	$S_{max}(M^a M)$	$S_{cnt}(\psi, M^a M)$
Audience: $M = M^a$	0.55 (0.54)	0.02 (0.05)
Composer: $M = M^c$	0.99 (0.99)	11.17 (11.23)

Table 3: Expected ψ -surprise for Unpaired training: $\psi = 0.1$. S_{cnt} is per 100 bits, composer accuracy 99% (99%).

Tables 2 and 3 report the surprise that the audience model M^a experiences on samples \hat{x}^c from composer (two-model sampling), as well as on samples \hat{x}^a from the audience itself (one-model sampling). For each model, the surprise numbers are averaged over 100 sampled sequences of 200 bits each. The results show that the audience model is much more surprised by examples sampled from the composer model, both in terms of maximum surprise S_{max} and average count of ψ -surprising events S_{cnt} . The sequences generated by the composer satisfy the opposite pattern \hat{x}^c shown on dark background in Table 1 with an accuracy of 98% or higher (accuracy numbers shown in the caption).

Delayed Resolution of Expectation

Delayed resolution of dissonance is one major tool composers use to play with the audience’s sense of expectation.

	Training patterns		Test
Audience	$x_1^a(0011)$	$x_2^a(1100)$	$x^a(0101)$
Composer	$x_1^c(00101)$	$x_2^c(11010)$	$\hat{x}^c(01001)$

Table 4: Audience & Composer examples for Delayed VoE.

To emulate this, we experimented with the dataset shown in Table 4 where the composer had to first violate the expectation (flip the consequent bit) and then satisfy it (follow with the expected bit). For example, by training on pattern $x_1^a(0011)$, the audience learns to compute a high expectation for the bit 1 whenever it follows the antecedent sequence 0, 0, 1. Thus, whenever the composer sees the antecedent 0, 0, 1, it also sees that the audience model has a high expectation for 1 to follow. By being trained on $x_1^c(00101)$, the composer learns that it should generate a 0 when the audience expectation is high, effectively violating the expectation, and then follow with the bit 1 expected by the audience.

Tables 5 and 6 show the results in the paired and unpaired training scenarios, respectively, using the same methodology as in the previous section. Here too the results show that the audience is much more surprised by examples sampled from the composer model, in terms of both maximum surprise and average frequency of ψ -surprising events.

Sampling model M	$S_{max}(M^a M)$	$S_{cnt}(\psi, M^a M)$
Audience: $M = M^a$	0.68 (0.67)	0.13 (0.14)
Composer: $M = M^c$	0.99 (0.99)	12.90 (11.71)

Table 5: Delayed ψ -surprise for Paired training, $\psi = 0.1$. S_{cnt} is per 100 bits, composer accuracy 99% (99%).

Sampling model M	$S_{max}(M^a M)$	$S_{cnt}(\psi, M^a M)$
Audience: $M = M^a$	0.54 (0.55)	0.00 (0.00)
Composer: $M = M^c$	0.99 (0.99)	11.98 (11.83)

Table 6: Delayed ψ -surprise for Unpaired training, $\psi = 0.1$. S_{cnt} is per 100 bits, composer accuracy 99% (98%).

Longer Patterns For all surprise scenarios from this section (delayed VoE) and the previous section (direct VoE), we also evaluated the composer model on longer patterns at test time, i.e. using an audience model trained on $x^a(01101)$, $x^a(111010)$, and $x^a(0101101)$. Even though at training time the composer had seen expectation patterns with only 4 bits, its performance on longer patterns was overall very similar with the performance reported in Tables 2 to 6, in terms of both surprise measures and accuracy. This can be seen as further evidence of the ability of the CA architecture with LSTMs to learn general VoE patterns.

Never-Ending Surprise

In this scenario, surprise is generated by violating high expectation at random. We first create a training dataset $D^a = \{\mathbf{o}_0\}$ for the audience that contains only a sequence of N random bits \mathbf{o}_0 . Let $D^c = \{\mathbf{o}_1, \mathbf{o}_2, \dots\}$ be a training dataset

for the composer containing two or more sequences that are similar to \mathbf{o}_0 but not exactly the same. Each sequence in D^c is generated by starting from \mathbf{o}_0 and randomly flipping bits, where at each position in the sequence the probability of flipping the bit is given by a Bernoulli distribution with mean $p = 1/N$. If $\text{Bernoulli}(p, N)$ is a sequence of N draws from this distribution, then each composer sequence \mathbf{o}_j can be seen as the element-wise exclusive-or between \mathbf{o}_0 and this random vector, i.e. $\mathbf{o}_j = \mathbf{o}_0 \oplus \mathbf{r}_j = \mathbf{o}_0 \oplus \text{Bernoulli}(p, N)$. For example, if $N = 10$, the two datasets could be as follows:

$$\begin{aligned} D^a &= \{\mathbf{o}_0 = \langle 0, 1, 0, 0, 1, 1, 0, 1, 0, 0 \rangle\} \\ &\quad \mathbf{r}_1 = \langle 0, 0, 1, 0, 0, 0, 0, 0, 0, 0 \rangle \\ &\quad \mathbf{r}_2 = \langle 0, 0, 0, 0, 0, 0, 1, 0, 0, 0 \rangle \\ &\quad \mathbf{r}_3 = \langle 0, 0, 0, 1, 0, 0, 0, 1, 0, 0 \rangle \\ D^c &= \{\mathbf{o}_1 = \mathbf{o}_0 \oplus \mathbf{r}_1 = \langle 0, 1, 1, 0, 1, 1, 0, 1, 0, 0 \rangle, \\ &\quad \mathbf{o}_2 = \mathbf{o}_0 \oplus \mathbf{r}_2 = \langle 0, 1, 0, 0, 1, 0, 0, 1, 0, 0 \rangle, \\ &\quad \mathbf{o}_3 = \mathbf{o}_0 \oplus \mathbf{r}_3 = \langle 0, 1, 0, 1, 1, 1, 0, 0, 0, 0 \rangle\} \end{aligned}$$

To train the composer model, first an audience model M^a is trained on D^a . Then the composer model $M^c \leftarrow (M^a, \mathbf{r})$ is trained on D^c , using as input the expectations computed by M^a for each example \mathbf{o}_j , as well as the random sample vectors \mathbf{r}_j . Upon training in this setting, the expectation is that the composer model will learn to violate the high expectations produced by the audience model only at the times specified by the random control vector \mathbf{r} . Thus, if the audience is trained on an arbitrary sequence \mathbf{o}_0 then, given a random control pattern \mathbf{r} , the composer should learn to compute mostly the element-wise exclusive-or between the two, i.e. $\mathbf{o} \approx \mathbf{o}_0 \oplus \mathbf{r}$ (deviations may happen due to the sampling done at each step in the sequence).

Algorithm 5 NEVERENDING(\mathbf{o}_0, N, p)

Input: An initial sequence of bits \mathbf{o}_0 .

Output: An infinite stream of bit sequences $\mathbf{o}_1, \mathbf{o}_2, \dots$

- 1: let $k = 0, D_0^c = \{\mathbf{o}_0\}$
 - 2: **for ever do**
 - 3: set $k = k + 1$
 - 4: train M_k^a on D_{k-1}^c
 - 5: let $\mathbf{r}_k \sim \text{Bernoulli}(p, N)$
 - 6: let $D_k^c = \{\mathbf{o}_k \sim M^c \leftarrow (M_k^a, \mathbf{r}_k)\}$
 - 7: **yield** \mathbf{o}_k
-

Once the composer model is trained, it can be used to generate an infinite stream of surprising sequences using Algorithm 5. The algorithm starts by initializing the set of compositions D_0^c with an input sequence \mathbf{o}_0 . This can be an arbitrary sequence of N bits, for example all zeroes. At every iteration of the never-ending generation loop, the current audience model M_k^a is trained on the previous set of compositions (step 4). Then a random control pattern \mathbf{r}_k is generated (step 5) and together with the current audience model M_k^a are used as input to the composer model M^c , which generates a new surprising sequence \mathbf{o}_k (step 6). An example output is illustrated below:

$$D_0^c = \{\mathbf{o}_0 = \langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle\}$$

$$\begin{aligned} &\quad \mathbf{r}_1 = \langle 0, 0, 0, 1, 0, 0, 0, 0, 0, 0 \rangle \\ D_1^c &= \{\mathbf{o}_1 \approx \mathbf{o}_0 \oplus \mathbf{r}_1 \approx \langle 0, 0, 0, 1, 0, 0, 0, 0, 0, 0 \rangle\} \\ &\quad \mathbf{r}_2 = \langle 0, 0, 1, 0, 0, 0, 0, 0, 1, 0 \rangle \\ D_2^c &= \{\mathbf{o}_2 \approx \mathbf{o}_1 \oplus \mathbf{r}_2 \approx \langle 0, 0, 1, 1, 0, 0, 0, 0, 1, 0 \rangle\} \\ &\quad \mathbf{r}_3 = \langle 0, 0, 0, 1, 0, 0, 0, 1, 0, 0 \rangle \\ D_3^c &= \{\mathbf{o}_3 \approx \mathbf{o}_2 \oplus \mathbf{r}_3 \approx \langle 0, 0, 1, 0, 0, 0, 0, 1, 1, 0 \rangle\} \\ &\quad \mathbf{r}_4 = \langle 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 \rangle \\ D_4^c &= \{\mathbf{o}_4 \approx \mathbf{o}_3 \oplus \mathbf{r}_4 \approx \langle 0, 0, 1, 0, 0, 0, 0, 0, 1, 0 \rangle\} \dots \end{aligned}$$

The composer LSTM was reduced to one layer with 2 neurons, and both models were trained for 5,000 epochs. The results in Table 7 show that the audience model experiences no surprise on samples from itself, whereas samples from the composer are very effective at eliciting surprise.

Sampling model M	$S_{max}(M^a M)$	$S_{cnt}(\psi, M^a M)$
Audience: $M = M^a$	0.00	0.00
Composer: $M = M^c$	0.99	1.65

Table 7: Expected Never-Ending ψ -surprise: $\psi = 0.1$, S_{cnt} is per 10 bits, composer accuracy 100%.

Relation to Previous Work

Itti and Baldi (2006; 2009) define the surprise of an agent M upon observing data D as the KL-divergence $KL(P(M|D)||P(M))$ between the posterior distribution of beliefs after the agent observes the data and its prior distribution of beliefs. This Bayesian definition of surprise is shown to be a good predictor of events that attract human attention in video frames. Macedo and Cardoso (2001) define the surprise of an event E as $1 - P_M(E)$. Given that a low probability alone cannot fully account for surprise (Teigen and Keren 2003), such as in models with uniform distributions, Macedo, Reizezein, and Cardoso (2004) refined the definition of surprise to also consider the most likely event E_h , i.e. $\log(1 + P(E_h) - P(E))$. Note that our definition of surprise naturally solves the uniform distribution dilemma by using a threshold that depends on the number of categories. Similar to (Macedo and Cardoso 2001), Horvitz et al. (2005) define surprising events to be those with low likelihood, e.g. 0.1 or less. They also go one step further and train a Bayesian network to forecast surprising events 30 minutes in advance for their traffic flow model JamBayes. In the context of evaluating creative designs, Maher, Brady, and Fisher (2013) identify surprising designs as outliers with respect to predictions based on features from previous designs.

Overall, these approaches were aimed at recognizing or forecasting surprise. To the best of our knowledge, the two-model architecture described in this paper is the first to address the task of producing surprising outputs by learning patterns of surprise from data. In terms of models that learn to generate surprising data, the most relevant work is Schmidhuber's Formal Theory of Creativity, summarized in (Schmidhuber 2012). There, the learning agent is entirely unsupervised and is expected to create novel and surprising data on its own, using a reinforcement learning algorithm

that rewards the agent when it generates data that helps it better compress its history of interactions with the environment. In contrast, our learning approach is data-driven in the sense that it is trained to minimize loss on data that is given to it, e.g. bit sequences. At the same time, like Schmidhuber’s creative agent, it does not require explicit supervision in terms of surprise, i.e. the composer is never told whether a particular event is surprising or not. The composer learns to surprise the audience only to the extent that the data provided to it is surprising for its model of the audience, which itself learns patterns of expectation from its own data.

There are also other two-model architectures, albeit designed for different purposes, such as the discriminator-generator model of generative adversarial nets (Goodfellow et al. 2014) or the student-teacher model used in the music theory learning system of (Yu and Varshney 2017).

Acknowledgments

We would like to thank Kristen Masada and Gordon Stewart for useful discussions on an earlier draft and the anonymous reviewers for their constructive feedback.

References

- Boden, M. A. 1991. *The Creative Mind: Myths and Mechanisms*. New York, NY, USA: Basic Books, Inc.
- Boulanger-Lewandowski, N.; Bengio, Y.; and Vincent, P. 2012. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *Proceedings of ICML’12*, 1–8.
- Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *NIPS*, 2672–2680.
- Grace, K., and Maher, M. L. 2015. Specific curiosity as a cause and consequence of transformational creativity. In *Proceedings of ICCG*, 260–267.
- Grace, K.; Maher, M. L.; Fisher, D.; and Brady, K. 2015. Modeling expectation for evaluating surprise in design creativity. In *Design Computing and Cognition*, 189–206.
- Hadjeres, G.; Pachet, F.; and Nielsen, F. 2017. DeepBach: a steerable model for Bach chorales generation. In *Proceedings of ICML’17*, 1362–1371.
- Hinton, G. E. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation* 14(8):1771–1800.
- Horvitz, E.; Apacible, J.; Sarin, R.; and Liao, L. 2005. Prediction, expectation, and surprise: Methods, designs, and study of a deployed traffic forecasting service. In *Proceedings of UAI’05*, 275–283.
- Huang, C.-Z. A.; Vaswani, A.; Uszkoreit, J.; Simon, I.; Hawthorne, C.; Shazeer, N.; Dai, A. M.; Hoffman, M. D.; Dinculescu, M.; and Eck, D. 2019. Music transformer. In *International Conference on Learning Representations*.
- Huron, D. 2008. *Sweet Anticipation: Music and the Psychology of Expectation*. MIT.
- Itti, L., and Baldi, P. F. 2006. Bayesian surprise attracts human attention. In *NIPS*. MIT Press.
- Itti, L., and Baldi, P. 2009. Bayesian surprise attracts human attention. *Vision Research* 49(10):1295 – 1306.
- Kingma, D. P., and Ba, J. L. 2015. Adam: A method for stochastic optimization. *International Conference on Learning Representations* 1–15.
- Larochelle, H., and Murray, I. 2011. The neural autoregressive distribution estimator. In *AISTATS*, 29–37.
- Macedo, L., and Cardoso, A. 2001. Modeling forms of surprise in an artificial agent. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 23.
- Macedo, L.; Reizezein, R.; and Cardoso, A. 2004. Modeling forms of surprise in artificial agents: empirical and theoretical study of surprise functions. In *Proceedings of the 26th Annual Meeting of the Cognitive Science Society*.
- Maher, M. L.; Brady, K.; and Fisher, D. H. 2013. Computational models of surprise in evaluating creative design. In *Proceedings of ICCG*, volume 147.
- Mas-Herrero, E.; Zatorre, R. J.; Rodriguez-Fornells, A.; and Marco-Pallarés, J. 2014. Dissociation between musical and monetary reward responses in specific musical anhedonia. *Current Biology* 24(6):699–704.
- Meyer, L. 1961. *Emotion and Meaning in Music*. University of Chicago.
- Nusbaum, E. C., and Silvia, P. J. 2011. Shivers and timbres: Personality and the experience of chills from music. *Social Psychological and Personality Science* 2(2):199–204.
- Oore, S.; Simon, I.; Dieleman, S.; Eck, D.; and Simonyan, K. 2018. This time with feeling: learning expressive musical performance. *Neural Computing and Applications*.
- Panksepp, J. 1995. The emotional sources of “chills” induced by music. *Music Perception: An Interdisciplinary Journal* 13(2):171–207.
- Sachs, M. E.; Ellis, R. J.; Schlaug, G.; and Loui, P. 2016. Brain connectivity reflects human aesthetic responses to music. *Social Cognitive and Affective Neuroscience* 11(6):884–891.
- Schmidhuber, J. 2012. A formal theory of creativity to model the creation of art. In McCormack, J., and dInverno, M., eds., *Computers and Creativity*. Springer-Verlag.
- Smolensky, P. 1986. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. MIT Press. chapter Information Processing in Dynamical Systems: Foundations of Harmony Theory, 194–281.
- Teigen, K. H., and Keren, G. 2003. Surprises: Low probabilities or high contrasts? *Cognition* 87(2):55–71.
- West, R., and Horvitz, E. 2019. Reverse-engineering satire, or “Paper on computational humor accepted despite making serious advances”. In *Proceedings of AAAI*, 1–8.
- Yannakakis, G. N., and Liapis, A. 2016. Searching for surprise. In *Proceedings of the Seventh International Conference on Computational Creativity*, 25–32.
- Yu, H., and Varshney, L. R. 2017. Towards deep interpretability (MUS-ROVER II): Learning hierarchical representations of tonal music. In *ICLR*.

Theoretical Framework for Computational Story Blending: From a Cognitive System Perspective

Taisuke Akimoto

Graduate School of Computer Science and Systems Engineering
Kyushu Institute of Technology
Iizuka, Fukuoka, Japan
akimoto@ai.kyutech.ac.jp

Abstract

The objective of this study is to design a general computational model of story creativity as the fundamental component of a cognitive system. In this paper, a theoretical framework for computational *story blending* is presented. This framework is inspired by cognitive and computational models of conceptual blending. Story blending, which is defined as composing a novel story by combining two input stories, is a fundamental principle of story creativity. Although the idea proposed in this paper has not been implemented yet, this study provides a theoretical basis for a computational modeling of story blending.

Introduction

Story creativity is the foundation of autonomous integrative artificial intelligence that can generate a contextual structure of the present situation, episodic memories, future goals and plans, the imaginations of the mental states of other persons, and hypothetical or fictional worlds. Meanwhile, conceptual blending theory, as proposed by Fauconnier and Turner (2002), characterizes the fundamental mechanism of human creative (but ordinary) thinking as the production of a novel concept by combining different familiar concepts. This cognitive theory has been applied to computational creativity studies, such as Eppe et al. (2018), Goguen and Harrell (2010), and Schorlemmer et al. (2014). This study seeks a general model of generative narrative cognition from a cognitive system perspective, whereby cognitive and computational models of conceptual blending are informative.

In this paper, a theoretical framework for computational *story blending* is proposed toward a general computational model of story creativity. Story blending is defined as composing a novel story by combining two input stories. In this context, a “story” refers to a mental representation of a narrative, whereas a “narrative” generally refers to information that is expressed in a communicational context.

Although the idea proposed in this paper has not been implemented yet, this study provides a theoretical basis for the computational modeling of story blending. A more detailed design and implementation will be presented in a future paper.

To illustrate the notion of story blending, Figure 1 shows an example of “narrative” blending by a non-expert human (a university student). A blended narrative (N3) was created by combining two given narratives (N1 and N2). This simple example contains various blending forms, e.g., merging temporal–spatial setting, replacing characters and their roles, and reconnecting the reason for a character’s action. Moreover, these operations are done in an integrated manner. Story blending refers to the cognitive process underlying this type of ability.

The rest of this paper is organized into five parts. First, previous related studies of narrative intelligence and conceptual blending are reviewed. Second, the significance of story blending is described. Third, several fundamental issues in computational modeling of story blending are discussed in three sections. Fourth, an architectural design of computational story blending is described. Finally, a conclusion and future prospective studies are presented.

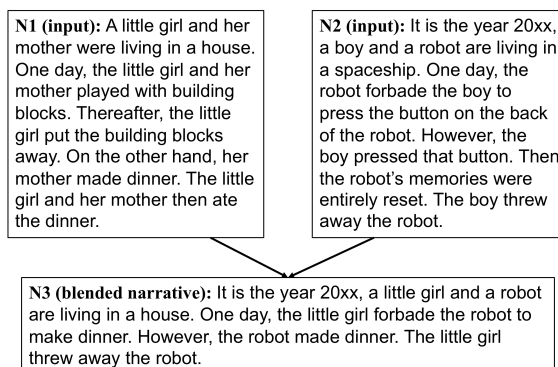


Figure 1. Example of “narrative” blending by a human subject.

Background

This section presents a review of previous related studies.

Computational Story Generation

Narrative generation is a challenging issue in artificial intelligence. In this context, the term “story” generation refers to the process of generating a content-level structure of a narrative, rather than an expression-level processes [in narratological terminology, a story or fabula refers to the content plane of a narrative, whereas a discourse or syuzhet refers to the expression plane (Prince 2003)].

There are several different but interrelated approaches to computational story generation. Some of the major approaches include the following:

- Planning-based approaches that model story generation as a simulation of the characters’ goal-directed actions in a specific world model (Meehan 1980; Riedl and Young 2010).
- Schematic approaches that formalize the generative structural knowledge of stories in the forms of story grammar (Pemberton 1989) and thematic structure (Bringsjord and Ferrucci 1999), among other forms.
- Case-based approaches that model story generation as the reconstruction of existing stories in various ways, including case-based reasoning (Turner 1994), retrieving possible next actions (Pérez y Pérez and Sharples 2001), and analogical reasoning (Riedl and León 2009; Ontañón and Zhu 2011).

Story blending can be regarded as a case-based approach.

Story in Cognitive Systems

Studies on cognitive systems or cognitive architecture are aimed at developing not only specific intellectual functionalities, but also general computational theories, models, frameworks, and systems for developing integrative intelligence. From a cognitive system perspective, a story or narrative can be considered as a universal form of knowledge, memory, or a mental representation of a subjective world.

Since the early years of artificial intelligence, researchers have focused on the roles of stories in a human intelligence. Their studies have led to several computational theories, including script (Schank and Abelson 1977) and dynamic memory based on memory organization packets (Schank 1982).

Recent studies have investigated the importance and universality of stories or narratives. For example, León (2016) proposed an architecture of narrative memory that focused on knowledge representation of episodic and procedural memories and narrative communication based on these memories. Samsonovich and Aha (2015) proposed a computational theory of goal reasoning based on a multilayered narrative structure and the notion of character. Akimoto (2018a) described the structures and functions of stories as attributes of an agent’s subjective world from four perspectives: a) constructing the contextual structure of the present situation; b) associating the past, future, and fiction

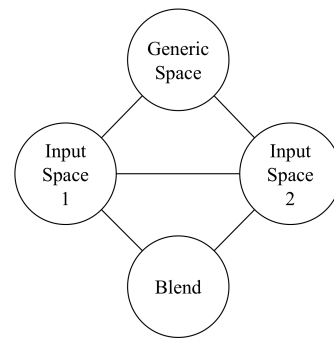


Figure 2. Simple illustration of conceptual blending.

with the present situation; c) imagining stories about others; and d) distinguishing between facts and fiction as metastory information.

For the above-mentioned reasons, an essential issue for cognitive systems is to achieve a general model of story creativity.

Computational Conceptual Blending

Conceptual blending theory (Fauconnier and Turner 2002) explains the fundamental mechanism of human creative thinking as the production of a novel concept by combining different familiar concepts. Figure 2 shows a simple illustration of conceptual blending. An *input space* (or an input mental space) refers to a small conceptual packet that provides source information for composing a *blend* (or a blended mental space). A blend is a new space produced by the combination of two or more input spaces. Here, a shared structure between input spaces, based on the cross-space mapping and counterpart connections, is captured into a *generic space*. This shared structure becomes part of the blend. However, the blend also contains other specific structures, including an emergent structure that is not directly projected from the input spaces.

Although conceptual blending was originally developed as a cognitive theory, several researchers have proposed computational models of conceptual blending. For example, Goguen and Harrell (2010) formalized conceptual blending by using algebraic semiotics as the basis for poetry narrative generation. In the COINVENT project (Schorlemmer et al. 2014; Eppe et al. 2018), the amalgam theory in case-based reasoning (Ontañón and Plaza 2010) was adapted into the core process of conceptual blending. Computational modeling of conceptual blending involves various subproblems.

Because conceptual blending generally has a huge solution space (i.e., possible combinations), it is necessary to formalize metrics for identifying “good” blends to prune the solution space. Eppe et al. (2018) introduced metrics for evaluating blends in terms of the amount of information, compression of structure, and balance of information. These metrics were defined based on the optimality principles of conceptual blending that were conceptually described by Fauconnier and Turner (2002). On the other hand, Confalonieri et al. (2018) introduced domain-specific values

from the perspective of audiences into the process of conceptual blending.

Constructing an adequate generic space is regarded as a key aspect of generating a consistent blend. As described by Besold (2018), analogical reasoning is a foundation of this process. From another perspective, Hedblom et al. (2016) adapted image schemas into the process of generalization or cross-space mapping as a representation of the abstract qualitative meaning of concepts.

The above-mentioned ideas and computational formulations of conceptual blending are applicable to story blending. However, story blending must deal with the content-level structures of stories, whereas computational conceptual blending treats the structures of general concepts. Goguen (2010) introduced a process called structural blending into poetry generation; this process focuses on composing a text-level structure. Computational conceptual blending also provides a basis for story creativity in inventing ideas of a unique character and an imaginative world setting. However, the primary focus of story blending is on manipulating an integrative and temporal structure that consists of concrete events and entities. This issue is a difficult aspect of story blending.

Story Blending in a Cognitive System

Story blending is a reasonable approach to a general model of story creativity for two reasons. First, producing new information and knowledge based on memory is an essential attribute of true autonomous intelligence. Second, stories can be regarded as integrative knowledge for composing a new story.

As described previously, story creativity is the common foundation for generating past memories; future expectations, predictions, goals, and plans; the contextual structure of the present situation; the imaginations of the mental states (i.e., theory of mind) of other persons; and hypothetical or fictional worlds. These aspects are necessary for an agent that autonomously interacts with its environment. In this context, the environment potentially includes all sorts of social and physical situations that an agent faces. Thus, interaction with the environment includes, for example, exploring a mountain, eating at a restaurant, communicating or cooperating with other persons (or agents) toward a goal, and creating an artistic work within the constraint of a specific genre.

Regarding the relationship to the environment, creative story generation can be classified as two types:

- *Adaptive* story generation: adaptation to an unfamiliar environment (e.g., the ability to generate a canonical story in a specific genre or to generate a story for acting appropriately at a restaurant).
- *Innovative* story generation: the challenge of making a change in the environment by producing a novel and valuable story or narrative (e.g., to generate a new style of story in or beyond a specific genre or to invent a new system of a restaurant).

Story blending aligns with both adaptive and innovative story generation. From the perspective of cognitive development, agents must adapt to new environments by using their own knowledge accumulated through previous experiences. The similarity between creativity and cognitive development is also described by Aguilar and Pérez y Pérez (2015). On the other hand, from the perspective of cultural development, an innovative story or narrative is essentially produced from the prior accumulation of social knowledge or narratives.

The next three sections discuss three fundamental issues for computational story blending: how to represent a story, how to deal with the structural complexity of a story, and what directs story generation.

Representation of a Story

From a cognitive system perspective, it is important to seek a general representational framework for a story as a uniform mental representation. However, this issue should be comprehensively addressed by considering various aspects of story cognition, including generation, understanding, analogy, blending, memory retrieval, embodiment or multimodality, and action–perception cycle. Hence, this study undertakes an exploratory design of a representation framework of a story from the perspective of story blending.

Stories and General Knowledge

To begin, it is important to distinguish between stories and general knowledge underlying stories (see Figure 3). The role of general knowledge here is to provide a common basis among different stories, even though every story is a unique item of information containing concrete events and entities arising at a time and a place. Narrative cognition generally requires various kinds of knowledge, including common sense knowledge. In story blending, categorical or attributive knowledge of words and relationships are especially required for structural comparison and manipulation of and between stories.

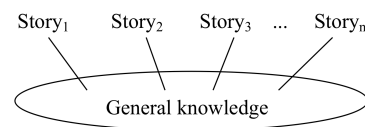


Figure 3. Stories and general knowledge.

Hierarchy of a Story

The fundamental structural units that form a story can be classified into four types, as follows:

- **Entity:** A character or object appearing in a story.
- **Event or State:** A character's action or stative information.
- **Relation:** A relationship between two entities or two events or states.
- **Time and place:** A temporal and spatial setting of a story or part of a story.

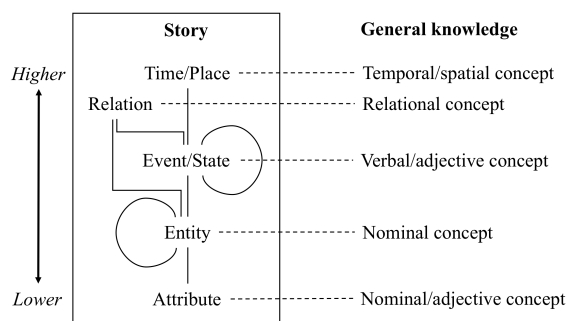


Figure 4. Hierarchy in a story and correspondence to general knowledge.

Thus, a story S is represented as a tuple $\langle N, V, R, T, P \rangle$ that consists of entities (N), events or states (V), relations (R), times (T), and places (P).

The relationships among these structural units can be interpreted as a hierarchical organization, as shown in Figure 4. In this hierarchy, the higher unit *contains* the lower unit. In particular, an entity contains attributes, an event or state contains entities as its arguments, a relation forms an integrative structure by containing two events or states or entities, and a time or place gives temporal or spatial setting, respectively, to the contained parts (events or states). The temporal order of events is also represented by anteroposterior relations. In addition, an aggregative event or state and entity is formed by containing two or more subevents or substates and subentities, respectively. For example, a “shortcake” can be seen as an aggregative entity containing “strawberry,” “whip cream,” and “sponge.” Similarly, the event “Lisa eats a steak in a restaurant” can be decomposed into several subevents.

Figure 4 also shows the corresponding general knowledge for each level of story element. Here, every story element is positioned as a unique instance of the corresponding concept.

Description Format

Because a story is composed of different types of structural units, designing a unified representation format for these units is a key issue in reducing the algorithmic complexity of story blending. Based on the hierarchy of a story, each unit can be represented by the same list format that consists of symbols for a head h and contained units c_i : (h, c_1, \dots, c_n) . Figure 5 shows an example of a simple representation of a story that is manually produced based on N2 in Figure 1.

How to Deal with the Structural Complexity of a Story

A story has a complex structure in which various representational elements are organized. In addition, story blending requires various semantic and structural processing. Hence, handling structural complexity is a difficult problem in computational story blending. Two approaches are introduced for addressing this problem: multiple abstraction and blend-centered perspective.

```
(n1:boy (name Jiro) (body small))
(n2:robot (sub n3 n4))
(n3:memory)
(n4:button (cause (reset memory)))
(v1:live (agent n1 n2) (location p1))
(v2:forbid (agent n2) (counter-agent n1)
  (object (press n4)))
(v3:press (agent n1) (object n4))
(v4:reset (object n3))
(v5:throw_away (agent n1) (counter-agent n2))
(r1:then v1 v2)
(r2:then v2 v3)
(r3:then v3 v4)
(r4:then v4 v5)
(r5:violation v3 v2)
(r6:cause v3 v4)
(r7:reason v4 v5)
(r8:partner n1 n2)
(p1:spaceship v1 v2 v3 v4 v5)
(t1:20xx v1 v2 v3 v4 v5)
```

Figure 5. Example of story based on N2 in Figure 1.

Multiple Abstraction

Abstraction is considered as a general issue in dealing with a complex problem or object in a computational system. According to Saitta and Zucker (2013), abstraction is an essential aspect of intelligence relevant to various cognitive activities, including problem solving, perception, analogy, categorization, language, and learning. Although the term “abstraction” has various meanings in various disciplines, the basic issues in abstraction can be organized into the following seven aspects: *simplicity*, *relevance*, *granularity*, *abstract or concrete status*, *naming*, *reformulation*, and *information content* (Saitta and Zucker 2013). Considering the first two aspects, simplicity means there is a general agreement that abstraction should reduce the complexity of tasks, and relevance means that abstraction is mainly supposed to capture the relevant aspects of problems, objects, or perceptions.

In story blending, abstraction can be regarded as the process of extracting manipulable partial information from a story from a *restrictive perspective*. This process is clearly different from generalization, which constructs a generic structure from input stories. The following are the various conceivable perspectives for story abstraction: “story-line” extracts the relational structure of events, excluding information on the entities; “story-world” extracts the relational structure of entities, excluding information on the events; “character perspective” extracts events and entities that are relevant to a specific character; and “temporal or spatial setting” extracts times or places from a story.

Based on the hierarchy of a story structure, abstraction can be defined as a top-down restriction or filtering of information to be extracted. In this process, the detailed contents of the extracted units may be parameterized as a variable or a category in general knowledge. Figure 6 shows an example of an abstraction (by hand) of the story in Figure 5 from the “story-line” perspective.


```

(v1:live (agent human#1 robot#1) (location
vehicle#1))
(v2:forbid (agent robot#1) (counter-agent
human#1) (object action#1))
(v3:press (agent human#1) (object button#1))
(v4:reset (object memory#1))
(v5:throw_away (agent human#1) (counter-agent
robot#1))
(r1:then v1 v2)
(r2:then v2 v3)
(r3:then v3 v4)
(r4:then v4 v5)
(r5:violation v3 v2)
(r6:cause v3 v4)
(r7:reason v4 v5)

```

Figure 6. Example of abstraction (by hand) of the story in Figure 5, from a “story-line” perspective.

Abstraction of stories precedes most processes in story blending, including comparison between stories, generalization of stories, and combinational integration of parts of stories. Moreover, story blending requires the combination of different abstractions from multiple perspectives.

Blend-centered Perspective

In previous studies on computational story generation, the generative process is generally modeled in the form of centrally controlled symbolic processing. However, from a long-term perspective, an emergentist approach is necessary for modeling complex cognitive processes, including story generation.

This approach is rooted in the work of Minsky (1986) that explains the mind as a type of distributed multi-agent system based on the collaborative activities of diverse simple agents. Inspired by this theory, Kokinov (1994) developed the DUAL cognitive architecture based on a distributed multi-agent system, whereby an agent refers to a small representational and procedural unit in a cognitive system. Akimoto (2018b) showed a conceptual-level theory of generative narrative cognition from an emergentist perspective. This theory posits that stories are fundamental agents that form a mind, and each story and its partial structures involve a power of self-organization.

Although implementing a fully distributed model of a generative story is still a distant goal, this study partially introduces an emergentist perspective. In particular, story blending is modeled as an internal process of the blended story to be generated. In other words, a blended story is an agent that generates its own structure.

Conceptual Diagram of Story Blending

By combining the above-mentioned approaches (multiple abstraction and blend-centered perspective), the diagram of conceptual blending (Figure 2) can be enhanced, as shown in Figure 7. Here, the blended story (S_b) composes its own structure by extracting information from two input stories (S_i and S_r).

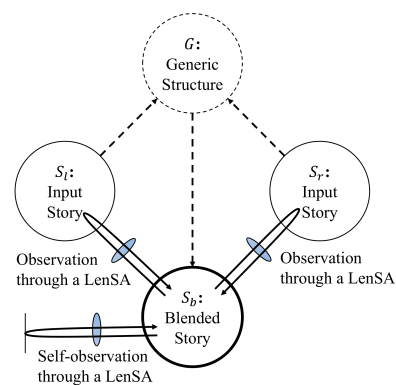


Figure 7. Conceptual diagram of story blending.

In this process, the blended story observes the inputs from a restrictive perspective. The term *lens of story abstraction (LenSA)* is introduced to refer to “observation equipment” for extracting an abstract structure from a story. More precisely, a LenSA corresponds to a function that extracts partial information from a story. The blended story uses LenSAs for gathering information from the input stories to generate its own structure. The blended story also observes its own structure through a LenSA to manage and direct the generative process.

The generic structure (G) refers to a common structure (which includes cross-story mapping) that emerges behind the two stories observed through a LenSA. Different generic structures can be constructed depending on the type of LenSA. In this case, because each story consists of unique instances, the commonality between stories is identified by categorical matching based on general knowledge.

Although the retrieval or recollection of input stories from memory is also an important issue, this topic is not included in this study, but will be addressed in future research.

What Directs Story Generation

Because the value of a story is highly dependent on the environmental context in which the story is used, defining absolute measures for identifying “good” stories seems inadequate when dealing with a general model of story creativity. Hence, this study takes a *relativistic* perspective by classifying criteria for directing story blending into *external* criteria based on the relationship with an environment and *internal* criteria based on the internal structure of a cognitive system. These criteria are described next.

External Criteria

External criteria for a story are defined based on values for oneself (the agent that produces the story), others (the receivers or users of the produced story or narrative), and societies in an environmental context. Although there are environmental dependencies, the basic types of external criteria can be classified along with the aforementioned notions of adaptive and innovative story generation, as follows:

- *Fitness* to an environment is determined from positive and negative feedback from that environment. Adaptive story generation needs to be modeled as an interactive system coupled with an environment and is primarily directed by fitness to that environment.
- *Effect* on an environment is determined based on a change in that environment (e.g., the effect on an audience's knowledge or worldview, creation of a new style or genre, and changes in cultural values). An effect on an environment is the essential condition for innovative story generation. However, the computational modeling of this criterion is a difficult problem.

Internal Criteria

Internal criteria provide only general conditions or driving forces for story generation. These criteria, which are independent of the environment, constitute the foundation of both adaptive and innovative story generation.

Producing *novelty* in story creation is assumed to be an essential condition of both adaptive and innovative story generation. The fundamental driving force for producing novelty can be formalized based on *difference* and *similarity* to the input or pre-existing stories in a cognitive system.

The agent and the environment may be viewed as developmental cognitive and social systems, respectively. A social system refers to a space of communication among two or more individuals under some form of constraints (e.g., a dialog between individuals, a communication within a team, or an artistic genre). Then, the notions of adaptive and innovative story generation can be reinterpreted. Adaptive generation is a trigger for a developmental change in the cognitive system itself. Similarly, innovative generation is a trigger for a developmental change in the environmental social system.

In both cases, difference is generally accepted as an essential condition of novelty. However, similarity is also necessary for organizing or anchoring new information (a story or narrative) into the relationship with pre-existing information. In other words, similarity is a constraint for continual development of both sides of the cognitive and social systems.

In the case of innovative story generation, difference and similarity to pre-existing information are determined not in a cognitive system, but in a social (environmental) system. However, if a cognitive system has acquired proficiency in that environment, a story's difference and similarity may be approximately simulated inside the cognitive system. For example, a cognitive system that has rich knowledge of a specific narrative genre will be able to compute the difference and similarity of a new idea based on its own memory.

In addition to difference and similarity, a fundamental condition for the internal structure of a story itself is also required. In particular, because a story is assumed to be an integrative structure that forms a mental world, the story must have *structural unity* or the coherence in the structure of the story. This attribute is the basis for composing the structure of the whole story.

In summary, the three internal criteria for directing story blending are presented as follows:

- **Difference and similarity:** A blended story must have both differences and similarities to the input stories. These criteria provide the driving force and constraint for achieving novelty.
- **Unity:** A blended story must have structural unity as the basic condition for the structure of the story.

Architectural Design of Story Blending

Based on the above-mentioned considerations, this section presents an architectural design for computational story blending. The objective of the proposed design from the three perspectives are presented next.

First, the proposed design focuses only on the process of blending the two given stories, without considering the process of retrieving stories from the memory.

Second, this study intends to present a fundamental principle of story creativity, instead of a specific application such as entertainment content generation. Hence, the proposed design of story blending aims at achieving environment-independence. From this stand point, the design focuses only on the aforementioned internal criteria and does not consider external criteria. Thus, the basic design objective is to develop a computational model that composes a blended story with structural unity and differences from or similarities to the given input stories.

Third, computational story blending involves various subproblems, including knowledge representation, abstraction, generalization, combination, similarity, difference, and unity. In each subproblem, there are various potential methods for implementing story blending. Hence, the proposed design aims at achieving an abstract theoretical framework for story blending by defining the basic representational and procedural elements and their relationships.

Structural Formulation

Basic representational elements of story blending, and their relationships, are illustrated by a hexangular diagram, as shown in Figure 8. These elements are defined as follows:

- S_l, S_r : Given (or retrieved) input stories.
- S_b : The blended story to be generated.
- A_l, A_r, A_b : Abstract structures extracted through a LenSA from S_l, S_r , and S_b , respectively.
- G_{lr}, G_{bl}, G_{br} : Generic structures constructed from A_l-A_r, A_b-A_l , and A_b-A_r , respectively. (G_{bl} and G_{br} have no counterpart in the original diagram of conceptual blending shown in Figure 2. These structures are used for calculating the differences and similarities between the blended and input stories.)
- $dif_{lr}, dif_{bl}, dif_{br}$: Numerical values representing the differences between each pair of stories observed through a LenSA, i.e., A_l-A_r, A_b-A_l , and A_b-A_r , respectively.
- $sim_{lr}, sim_{bl}, sim_{br}$: Numerical values representing the

similarities between each pair of stories observed through a LenSA, i.e., A_l-A_r , A_b-A_l , and A_b-A_r , respectively.

- $unity_b$: A numerical value representing the structural unity of S_b .

These elements, excluding input stories, are dynamically generated and rewritten through the generative process.

Procedural Formulation

Figure 9 illustrates the procedural framework of story blending. From a blend-centered perspective, the procedure of story blending is designed based on the internal processes of a blended story. However, the LenSAs, generalization, and combination can be considered as automatic processes that generate abstract, generic, and combinational structures, respectively. Hence, these elements are positioned as external processes. Overall, a blended story generates its own structure (S_b) from two input stories (S_l and S_r) with general knowledge by using functions of the LenSAs, generalization, and combination.

Basic Functions

In the framework shown in Figure 9, the following procedural elements are defined as functions:

- $LenSA(X, S_p)$: Extracting an abstract structure A_p from a story. Here, the type of LenSA (e.g., story-line or story-world) is specified by X , which is determined by the self-manager part as described later.
- $generalization(A_p, A_q)$: Constructing a generic structure G_{pq} , including cross-space mapping between structural units, from two abstract structures.
- $combination(A_p, A_q, G_{pq})$: Generating a set of candidate combinational structures $C = \{c_1, \dots, c_n\}$ of two abstract structures. A combinational structure is also a partial structure of a story that is constructed by selective integration of two abstract structures.
- $difference(A_p, A_q, G_{pq})$: Calculating dif_{pq} .
- $similarity(A_p, A_q, G_{pq})$: Calculating sim_{pq} .
- $unity(S_p)$: Calculating $unity_p$.

Self-Manager

The self-manager controls its own generative process. This iterative sequence of processes involves extracting abstract structures from the input stories, combining the abstract structures, and integrating a combinational structure into the blended story until the blending is completed. Although the detailed design will be performed in future work, a tentative framework of the blending process is presented as follows:

Step 1: Selection of a LenSA. The self-manager chooses a LenSA based on similarities and differences. Various selection strategies are conceivable, such as a similar aspect between inputs (higher sim_{lr}), a different aspect between inputs (higher dif_{lr}), and lack of information in the current blend structure (higher dif_{bl} and dif_{br}). When a LenSA is chosen, abstract structures (A_l and A_r), a generic structure (G_{lr}), and a set of combinational structures (C) are automatically generated.

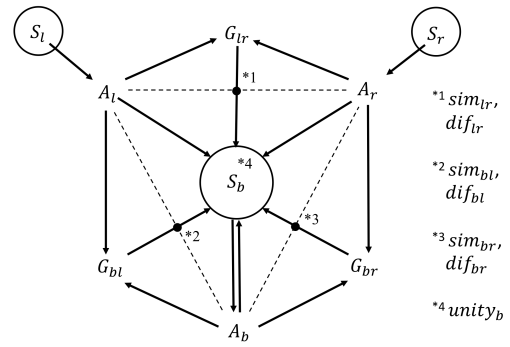


Figure 8. Diagram of story blending.

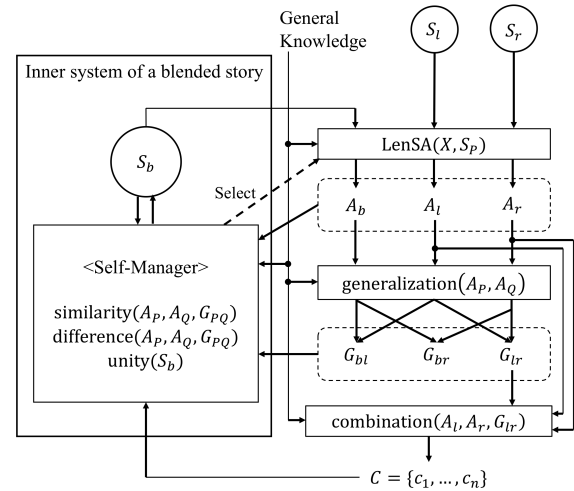


Figure 9. Procedural framework of story blending.

Step 2: Selection of a combinational structure. By assuming each combinational structure (c_i) as the abstract structure (A_b) of the blend, the self-manager chooses a combinational structure that has *higher* and *balanced* values of sim_{bl} , sim_{br} , dif_{bl} , and dif_{br} , in total.

Step 3: Integration. The self-manager integrates the selected combinational structure into the blended story. Structural adjustment will also be required here.

Step 4: Completion judgment. The self-manager observes the blended structure and judges whether to capture additional information from the input stories (i.e., return to Step 1) or to proceed to the final adjustment process (Step 5).

Step 5: Final adjustment. The self-manager completes the blended structure and content to increase $unity_b$.

Concluding Remarks

In this study, story blending was presented as a fundamental principle of story creativity in a cognitive system. From this perspective, three basic issues were discussed. First, a representational framework of a hierarchical story structure was presented. Second, two approaches for managing

structural complexity in a story (i.e., multiple abstraction and blend-centered perspective) were introduced. Third, the criteria of directing story generation were classified into external criteria (based on the relationship with an environment) and internal criteria (based on the internal structure of a cognitive system). This study especially focused on the latter and stated three essential internal criteria: differences and similarities to existing (input) stories and structural unity of the blended story. Based on these concepts, an architectural design of computational story blending was presented.

The next stage of this study will create algorithms of the system elements, including abstraction, generalization, combination, and calculations of difference, similarity, and unity. Particularly, there are two primary challenges in the future. The first one is to formulate the mechanism of abstracting a story through multiple structural perspectives. This mechanism will be a basis for not only story blending, but also broad aspects of story cognition. The second challenge is to develop a general model for combining two (abstracted) stories via their generalization.

Acknowledgments

This work was supported by The Telecommunications Advancement Foundation and JSPS KAKENHI Grant Number JP18K18344.

References

- Aguilar, W., and Pérez y Pérez, R. 2015. Dev E-R: A computational model of early cognitive development as a creative process. *Cognitive Systems Research* 33:17–41.
- Akimoto, T. 2018a. Stories as mental representations of an agent's subjective world: A structural overview. *Biologically Inspired Cognitive Architectures* 25:107–112.
- Akimoto, T. 2018b. Emergentist view on generative narrative cognition: Considering principles of the self-organization of mental stories. *Advances in Human-Computer Interaction* 2018:6780564.
- Besold, T. 2018. The relationship between conceptual blending and analogical reasoning. In Confalonieri, R. et al. eds., *Concept Invention: Foundations, Implementation, Social Aspects and Applications*. Cham: Springer. 133–151.
- Bringsjord, S., and Ferrucci, D. A. 1999. *Artificial Intelligence and Literary Creativity: Inside the Mind of BRUTUS, a Storytelling Machine*. NJ: Lawrence Erlbaum.
- Confalonieri, R., Plaza, E., and Schorlemmer, M. 2018. Computational aspects of concept invention. In Confalonieri, R. et al. eds., *Concept Invention: Foundations, Implementation, Social Aspects and Applications*. Cham: Springer. 31–67.
- Eppe, M., Maclean, E., Confalonieri, R., Kutz, O., Schorlemmer, M., Plaza, E., and Kühnberger, K. U. 2018. A computational framework for conceptual blending. *Artificial Intelligence* 256:105–129.
- Fauconnier, G., and Turner, M. 2002. *The Way We Think: Conceptual Blending and the Mind's Hidden Complexities*. NY: Basic Books.
- Goguen, J. A., and Harrell, D. F. 2010. Style: A computational and conceptual blending-based approach. In *The Structure of Style: Algorithmic Approaches to Understanding Manner and Meaning*. Berlin: Springer-Verlag. 147–170.
- Hedblom, M. M., Kutz, O., and Neuhaus, F. 2016. Image schemas in computational conceptual blending. *Cognitive Systems Research* 39:42–57.
- Kokinov, B. N. 1994. The DUAL cognitive architecture: A hybrid multi-agent approach. In *Proc. 11th European Conference on Artificial Intelligence*, 203–207.
- León, C. 2016. An architecture of narrative memory. *Biologically Inspired Cognitive Architectures* 16:19–33.
- Meehan, J. R. 1980. *The Metanovel: Writing Stories by Computer*. NY: Garland Publishing.
- Minsky, M. 1986. *The Society of Mind*. Simon & Schuster.
- Ontañón, S., and Plaza, E. 2010. Amalgams: A formal approach for combining multiple case solutions. In *Lecture Notes in Computer Science (Proc. ICCBR 2010)*, LNCS 6176:257–271.
- Ontañón, S., and Zhu, J. 2011. The SAM algorithm for analogy-based story generation. In *Proc. 7th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 67–72.
- Pemberton, L. 1989. A modular approach to story generation. In *Proc. 4th Conference on European Chapter of the Association for Computational Linguistics*, 217–224.
- Pérez y Pérez, R., and Sharples, M. 2001. MEXICA: A computer model of a cognitive account of creative writing. *Journal of Experimental & Theoretical Artificial Intelligence* 13(2):119–139.
- Prince, G. 2003. *A dictionary of narratology (revised ed.)*. NE: University of Nebraska Press.
- Riedl, M. O., and León, C. 2009. Generating story analogues. In *Proc. 5th Artificial Intelligence and Interactive Digital Entertainment Conference*, 161–166.
- Riedl, M. O., and Young, R. M. 2010. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research* 39:217–267.
- Turner, S. R. 1994. *The Creative Process: A Computer Model of Storytelling and Creativity*. NJ: Lawrence Erlbaum.
- Riesbeck, C. K., and Schank, R. C. 1989. *Inside Case-Based Reasoning*. NJ: Lawrence Erlbaum.
- Saitta, L., and Zucker, J.-D. 2013. *Abstraction in Artificial Intelligence and Complex System*. NY: Springer.
- Samsonovich, A. V., and Aha, D. W. 2015. Character-oriented narrative goal reasoning in autonomous actors. In *Goal Reasoning: Papers from the ACS Workshop*, GT-IRIM-CR-2015-001:166–181.
- Schank, R. C., and Abelson, R. P. 1977. *Scripts, Plans, Goals, and Understanding: An Inquiry into Human Knowledge Structures*. NJ: Lawrence Erlbaum.
- Schank, R. C. 1982. *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*. NY: Cambridge University Press.
- Schorlemmer, M., Smaill, A., Kühnberger, K.-U., Kutz, O., Colton, S., Cambouropoulos, E., and Pease, A. 2014. COINVENT: Towards a computational concept invention theory. In *Proc. 5th International Conference on Computational Creativity*, 288–296.

Duets Ex Machina:

On The Performative Aspects of “Double Acts” in Computational Creativity

Tony Veale, Philipp Wicke and Thomas Mildner

School of Computer Science, University College Dublin, Ireland.

Abstract

We humans often compensate for our own weaknesses by partnering with those with complementary strengths. So fiction is full of characters who complete each other, just as show-business thrives on successful double acts. If it works for humans, then why not for our machines? The comparative strengths and weaknesses of different CC systems are well-documented in the literature, just as the pros & cons of various technologies or platforms are well known to the builders of these systems. A good pairing does more than compensate for the weaknesses of one with the strengths of another: it can find value in disparity, and deliver results that are beyond the reach of either partner alone. Here we consider the pairing of two CC systems in the same thematic area, a speech-based story-teller (with *Alexa*) and an embodied story-teller (using a *NAO* robot). Working together, these two compensate for each other’s weaknesses while creating something of comedic value that neither has on its own.

In It Together

The mythology of human creativity often paints a romantic image of the solitary creator, toiling against the status quo to fulfil a singular vision. But our creativity narratives also prize the results of successful partnerships. One can list a long line of inspired double acts, from Crick & Watson – or, indeed, Holmes & Watson – to Lennon & McCartney, in which a duo’s differences count as much as what they share. If good partners learn to overcome their differences, creative partners learn to *exploit* their differences, and no where is this truer than in the classic comedy double act.

Henri Bergson (1911) has argued that mechanical rigidity lies at the root of all comedy. We become risible when we are reduced to predictable machines and act unthinkingly in the pursuit of conformity. Yet Freud (1919) has also argued that when machines take on human characteristics, such as the semblance of free will, they appear *uncanny* or *unheimlich*, sources of terror rather than agents of comedy. Our CC systems can be nudged either way on this continuum of the *canned* to the *uncanny*, to play their presumed stiffness for laughs or to transcend this rigidity by acting unpredictably. Most comedy double acts do both, with one partner serving as a defender of conformity, the other as an

agent of chaos. In their interactions we see glimpses of the *relief theory* of humour as espoused by Lord Shaftesbury (1709): the free agent shows a nimbleness of spirit and an ability to break free of its constrainer, the rigid partner. The latter looks stiff and inadequate, following Bergson, while the former looks graceful and agile, following Shaftesbury, so both theories together give us twice the reason to laugh. Famous comedy acts from Stan Laurel and Oliver Hardy to Bob Hope and Bing Crosby to Dean Martin and Jerry Lewis all worked in solo acts first, as singers, actors and comics, before coming together to reap the benefits of their obvious friction and complementarity (see e.g., Epstein, 2005).



Figure 1. The Walkie-Talkie double act of NAO and Alexa.

When friction sparks comedy, each part of the duo acts as a tacit rebuke to the other; the straight guy is *too* rigid, and the funny guy is *too* unpredictable. This is not simply a matter of how material is divided up and performed, but an issue of substance in the material itself. For laughter can be wrung from a meta-critique of the act’s artifice, as when a ventriloquist’s dummy says to its human partner, “Why is

it that every time I shout, I get sprayed with *your* spittle?” A ventriloquist and his dummy are two roles played by one performer, which an audience willingly sees as two agents. Each, however, represents a different part of the psyche of a single idealized performer, the *super-ego* (ventriloquist) and the *id* (dummy). One works to keep the other in check, and fails, but it is in this failure that the comedy takes root. Computationally, the fact that one CC system works as two gives us a convenient abstraction for a comedic double act. A single system, coordinated using backstage computation, controls two agents of conflicting temperament that create comedy through their interactions on the same shared task.

The rest of the paper puts flesh on our scheme, in which a *NAO* robot and an *Amazon Echo* are used to implement a story-telling double act (see Figure 1). We show how their complementary strengths and weaknesses are exploited to make a virtue of failings that would be nigh on intolerable in one alone. Our aim is to turn each platform into an agent with its own personality, rather like the bickering droid duo R2D2 and C3PO in *Star Wars*. The next section presents a story-telling skill for the *Echo*'s speech-driven *Alexa* front-end, before an embodied, *NAO*-based robot story-teller, for the same space of computer-generated stories, is described. This story space is built using *Scéalextric* (Veale, 2017), a story-generation CC system ideally suited to the creation of shaggy dog tales that put familiar faces in comical settings. We present an advance to *Scéalextric* that imposes a global shape on its plots and supports the generation of narratives of more than two key characters. These tales are performed by a double-act, named *Walkie Talkie*, of *Alexa* and a *NAO* robot, in which *Alexa* narrates a tale as the *NAO* embodies its actions. Coordinating their interactions is a blackboard architecture that obviates the need for any overt communication, yet we focus here on the ways in which their joint performance is built upon the interplay of the spoken and the physical. We show how the clear-spoken *Alexa* can act as the straight guy while the clownish *NAO* can be her foil. The paper concludes with a discussion of related work and a map of future directions for the *Walkie Talkie* double act.

Alexa in Storyland

Though the browser was once our principle means of web access, and a convenient platform for offering CC systems as services, the advent of devices such *Amazon Echo* and *Google Home* has given CC systems an alternate route into our homes. Consider *Alexa* (Amazon, 2019) a speech-activated ‘genie’ that answers our questions, fetches our data and controls our music, lighting, heating and more. *Alexa*'s repertoire of skills is easily extensible, allowing developers to add new ‘skills’ for the delivery of content that may well be machine-generated. So, in addition to fetching factoids, weather updates, recipes and canned jokes, *Alexa* can be extended to create riddles and poems, and even stories, on demand. Yet, since story-telling is an art, a narrative ‘skill’ for such a device must exploit all of the affordances, and sidestep all the impediments, of the technology concerned.

Each *Alexa* skill is opened with a voice command, as in “*Alexa*, open the narrator.” Once inside an open skill, users

may use a variety of pre-defined speech patterns to achieve a given end. Our story-telling skill, *The Narrator*, can be requested to tell stories on a specific theme, as in “*Alexa*, tell me a story about love” or “*Alexa*, tell me a *Star Wars* story.” Once a topic is extracted, the skill fetches an apt story from a large pool of pre-generated tales. *Alexa* skills may call on a variety of Amazon Web Service components, such as an AWS database, to store the knowledge / data of a CC system, so that creative artifacts can be generated by the skill on the fly. However, as each skill must package its response in a fixed time (8 seconds) before the current task is aborted, we prefer to use *Scéalextric* and the *NOC list* (see Veale, 2016) to pre-generate hundreds of thousands of stories in advance, storing each with appropriate indexes to facilitate future thematic retrieval. The step function of the AWS pricing model is rather steep, and if one is not careful about data usage a skill can jump from costing nothing at all to costing hundreds of dollars per month. Yet, as shown in Veale & Cook (2018), pre-built spaces of content offer a clean and efficient approach to the separation of creation, curation, selection and delivery tasks. In our case, we opt to store our large story space on the Web, and *Alexa* dips into different parts of this space using topic-specific URLs.

The *Alexa intent model* is powerful and flexible, but can seem counter-intuitive from a conventional programming perspective. Accommodations must be made to repackage a CC system as an *Alexa* skill, and the process is not unlike building a ship inside a bottle. Yet the payoffs are obvious: *Alexa* has excellent speech comprehension and generation capabilities for a consumer device; the former is robust to ambient noise while the latter sounds natural, if prim, so in a story-telling double act, *Alexa* is destined to play the role of straight guy. Her formal disembodied voice reminds us of *HAL 9000* and any number of sci-fi clichés about rigid machines, making *Alexa* a natural fit for Bergson's theory.

Her rigidity extends to a lack of reentrancy in how skills are executed. *Alexa* retrieves whole stories from her online story space, choosing randomly from tales that match the current theme to produce a single, composite speech act for a narrative. Users can interrupt *Alexa* to stop one story and request another. but *Alexa* cannot segment a narrative into beats of a single action apiece, and articulate each beat as a distinct response to the user. That would require her to re-entrantly jump in and out of her narration intent, at least if she needs to execute other tasks between beats. This makes uninterrupted story-telling difficult to align with the actions of parallel performers, as choreography demands chunking, communication and reentrancy. This is not a problem when *Alexa* works alone; she simply narratives her chosen story in a single continuous speech act. But when she must work with a partner, such as an embodied robot, this double act requires her to articulate the story one beat at a time, and wait for a prompt from a human – such as “yes,” “go on,” “uh huh,” “really?” or “then what?” – to proceed. In the gap opened by this interaction, *Alexa* is free to communicate with her partner and cue up the partner's enacted response.

For long stories – and our improvements to *Scéalextric* produce tales of multiple characters and many beats, as we

describe in a later section – the need for an explicit prompt between each beat is an onerous one. Without this prompt, *Alexa* can do little, and her partner will also lack the cue to perform, bringing their double act to a standstill. However, as with human double acts, this rigidity of form is itself an opportunity for meta-comedy. When *Alexa* becomes stuck, as when it fails to receive or perceive a prompt, her partner offers a wry comment on the situation. These meta-actions constitute the double act’s shared mental model (Fuller & Magerko, 2010), perched above its content-specific *domain* model, allowing an act to be more than the sum of its parts. This setup is not so different to a human ventriloquist with an insolent dummy: while *what* is said is vitally important, *how* it is said and enacted is a source of humorous friction.

Apocalypse NAO

Alexa has a voice but no body. The *NAO* has both a body and a voice, but the limitations of the latter often struggle to transcend the former. Although the *NAO*’s capacity for physical movement is a major selling point, its gestures can be so noisy as to dominate its twee vocalizations. Moreover, *NAO*’s processing of speech is rather limited in comparison to *Alexa*’s, and frequently forces its human interlocutors to vehemently repeat themselves on even short commands. So a pairing of *Alexa* & *NAO* makes sound technical sense for a language-based task like storytelling, since *NAO*’s utility as an embodied storyteller has already been demonstrated by Pelachaud *et al.* (2010) and Wicke *et al.* (2018a,b). As the latter uses the *NAO* to tell computer-generated stories, we use that work here as a foundation for our CC system.

With a humanoid body offering 25 degrees of freedom, a *NAO* can pantomime almost every action in a story. Wicke *et al.* (ibid) built a mapping of plot verbs to robot gestures, so that their robot has an embodied response to each of the 800 verbs in the underlying story-generator, the *Scéalextric* system of Veale (2017). Two variants of the storyteller are presented. Wicke *et al.* (2018a) describe how pre-generated *Scéalextric* stories are selected at random and enacted with a combination of speech – to articulate each beat of a story – and gesture, to simultaneously pantomime the action. The chosen story is retrieved using a vocal cue from the user, who provides a topic index such as “love” or “betrayal.” In Wicke *et al.* (2018b), the user exerts more control over the shape of the story. In this variant, the robot uses the causal graph connecting *Scéalextric* actions to generate questions that require users to probe their own experiences and offer yes/no answers in response. The answers allow the robot to navigate the space of *Scéalextric* stories to build a tale that is a bespoke fit to the user’s tastes. However, each variant works solely at the content-level, using a domain model to map directly from generic story verbs to robot capabilities.

A storyteller transcends its domain model – its model of what constitutes a story – whenever it shows awareness of itself as a teller of the tale. This is storytelling taken to the meta-level, in which a teller acknowledges its dual status as a protagonist who *lives* the tale via physical actions and an omniscient narrator who relates the tale via speech acts. The domain model ensures the effective communication of

character-to-character relations, whilst the meta-model is responsible for *teller-to-audience* relations, as well as, for a double act, *teller-to-teller* relations. Of the two, the domain model is the most immediate, and has received the greatest attention from researchers. Pantomime is the obvious basis for a robot’s domain model, but tellers can take an abstract view of events without wandering into the meta-level. For instance, following Pérez y Pérez (2007), a teller can track the disposition of characters to each another. In *Scéalextric* stories of just two characters using a finite number of plot verbs (approx. 800), it is feasible to mark each action as to whether it tends to promote closeness or distance. So, *love*, *respect* and *trust* are verbs that bring closeness, while verbs such as *insult*, *betray* and *suspect* each increase distance. A robot teller can assign each character to a distinct hand, so that as the story progresses, the horizontal movement of its hands conveys the conceptual distance between characters.

The meta-model of a storyteller recognizes that there are many ways to exploit the domain model to convey a story. Montfort’s *Curveship* system for interactive fiction (2009) shows how a meta-model can alter the dynamic of a tale by opting to focalize one character over another, or by switching between narrators and rendering styles. Montfort *et al.* (2013) use a blackboard framework to integrate their storytelling system with a metaphor generator whilst exploiting the affordances of the *Curveship* meta-modal. The domain model is responsible for *in-world* reasoning about a story, so only the meta-model acknowledges the existence of the audience, other performers, and the artifice of the process. Often, however, the distinction between domain- and meta-models is a subtle one. To an audience, there may be little difference between a robot pantomiming the reactions of other characters to a specific act – for example, by reacting with surprise or the disappointed shake of a bowed head – and gesturally signifying its own reaction as a narrator. In the final analysis it matters little if the audience can tell the domain- and meta-models apart, as long as the story is told with aplomb. Nonetheless, a meta-model works best when it augments rather than supplants the domain model. When an agent is aware of its role, it can act as a character or as a narrator or even as an audience member if it serves the tale.

The meta-model is dependent on the domain model for its insights into the story, to e.g., determine which parts are tense and dramatic or loose and comedic. With such insight an embodied teller can react appropriately to its own story, by feigning shock, joy or even boredom in the right places. In a double-act, these reactions must be coordinated across performers, so that they are seen by the audience not just as responses to the story but to each other. For instance, if the embodied agent (e.g., *NAO*) pretends to sleep at a certain point, the speech agent (e.g. *Alexa*) may join the pretence and wake it up with a rebuke or a self-deprecating remark. Each performer will have its own domain model suited to its own modality, and its own meta-model. But each will need to share a joint meta-model to permit coordination.

It’s worth noting that in addition to the *NAO*’s physical affordances for pantomime, it also offers some support for vocal mimicry. So while its built-in voice is twee, the robot

permits one to upload arbitrary sound files and recordings, making the use of 3rd-party voice synthesis tools (such as those offered by IBM Watson) a viable option. We draw on this service when we want *NAO* to communicate directly with *Alexa* and to have its voice prompts understood as commands, since *Alexa* does not react to the *NAO*'s normal speaking voice. It can also be used to associate a different speaking voice with different meta-model functions, from making wisecracks about the current story to making fun of the audience to poking fun at the system's developers. A key use of this ability is the coordination of meta-models. The *Alexa* narrator articulates each beat of the story before waiting for the *NAO* to respond in an embodied fashion. Since neither knows how long the other will take, they use conversation (of a sort) to align their own private models.

Skolem Golems and Scéalextric

The *Scéalextric* system of Veale (2017) offers an open and extensible approach to story-generation that has sufficient knowledge to build both the domain- and meta-models. A plot in *Scéalextric* is built from plot triples, each of which, in turn, comprises of a sequence of three plot verbs. In all, *Scéalextric* provides semantic support for 800+ plot verbs, by indicating e.g. how each verb causally links to others, or how each verb can be idiomatically rendered in a final text. Each verb is assumed to link the same two protagonists, in a story of just two characters overall. It balances this limitation by exploiting a vivid cast of familiar fillers for these two roles, drawing on the *NOC list* of Veale (2016) to provide detailed descriptions of over 1000 famous characters. Veale (2017) reports empirical findings as to the benefit of reusing familiar faces in shaggy-dog tales, noting that readers rate such tales as more humorous and more eventful. Yet the shagginess of these tales is exacerbated by the way that triples are connected, end-over-end, to generate what amounts to a random walk in the causal graph of plot verbs. Though *Scéalextric*'s plot graph has over 3000 edges connecting its 800+ verbs with arcs labeled *so*, *then*, *and*, *but*, the resulting stories exhibit local coherence at the expense of global shape. Its tales meander, and lack a clear purpose.

The limitations of *Scéalextric* as a domain model need to be remedied if a rich meta-model is to be built on top of it. A story of just two characters does not afford much variety for even a single performer to leverage, much less a double act, whilst the lack of coherent sub-plots that return to the main story trunk also reduces the potential for play at the meta-level. We remedy both deficiencies with a new kind of triple that is designed to be expanded recursively, into a plot tree, rather than additively into a rambling plot line. So rather than connecting plot triples end-to-end, our approach will expand these new triples via recursive descent from a single starting triple that gives each story its global shape.

Consider how *Scéalextric* (Veale, 2017) defines and uses its triples. Suppose TUV, VWX and XYZ are triples made from the plot verbs *T*, *U*, *V*, *W*, *X*, *Y* and *Z*. Then each verb is assumed to take two implicit character slots, α and β , which are later filled with two specific characterizations drawn from the NOC list. So the triple XYZ is in fact the

sequence $\langle \alpha X \beta \rangle \langle \alpha Y \beta \rangle \langle \alpha Z \beta \rangle$. Triples are connected end-over-end, with the last verb of one matching the first verb of the next. In this way, TUV, VWX and XYZ can be combined to construct the story TUVWXYZ. The causal graph provides a labeled edge between any two plot verbs that are linked by at least one triple; the label set is $\{so, then, and, but\}$. So if given the starting verb *T* and the ending verb *Z* in advance, a system can search the graph of causal connections to find a story of a stated minimum size that starts with the action $\langle \alpha T \beta \rangle$ and ends with $\langle \alpha Z \beta \rangle$.

In our augmentations to *Scéalextric*, we add a range of triples of the form T-X-Z, where *T*, *X* and *Z* are plot verbs and the hyphen – denotes a point of recursive expansion. Thus, T-X and X-Z admit additional content to link *T* to *X* and *X* to *Z*. This content is inserted as further triples, such as XYZ (to link *X* and *Z*) or T-V-X. The latter links *T* to *X* via another recursive triple that requires expansion in the gap from *T* to *V* and from *V* to *X*. The nonrecursive triples TUV and VWX can fill these gaps to yield a complete plot, TUVWXYZ. Notice how the existing stock of *Scéalextric* triples is reused, not replaced, and simply augmented with new triples that operate top-down rather than left to right. A subset of the new recursive triples are marked as suitable for starting a story; these give each plot its global shape. At present we designate over 200 recursive triples to be story starters, but these can be adjoined in a left-to-right fashion (as in the original *Scéalextric*) to create higher-level story shapes. Thus, the triples A-J-T and T-X-Z may be adjoined to create a story that starts with action *A* and ends with *Z*.

For stories with just two characters a generator need not worry about under-using a character, especially if each plot verb – as in *Scéalextric* – assumes the participation of both. The introduction of arbitrarily many additional characters can enrich a narrative greatly, but at the cost of complexity. All characters must be kept in play, and not forgotten even when they are not participating in the current action or sub-plot. A sub-plot is a story path that diverges from the main trunk of the narrative and rejoins it at a later time. Consider a story in which character α assaults character β . A viable sub-plot involves α being investigated for the assault by a third character γ that fills the role of detective. The sub-plot may recursively draw in a fourth character, a lawyer for α , which then necessitates the introduction of a lawyer for β . When the sub-plot ends and the plot rejoins the main trunk, these additional characters can be forgotten, but not before.

We add a capacity for additional temporary characters to *Scéalextric* via skolemization. If β is a character, β -spouse denotes the love interest of β in $\langle \alpha seduce \beta$ -spouse \rangle , so whatever NOC character is chosen for β , a relevant NOC character is also chosen to fill β -spouse (e.g. Bill Clinton for Hillary Clinton). Other skolem functions include *friend*, *enemy*, *partner*, and each exploit the NOC in its own way. α -friend, for instance, is a character with a high similarity to the filler for α (e.g. Lex Luthor for Donald Trump), while α -partner is instantiated with a character of the same group affiliation in the NOC (e.g., Thor for Tony Stark, as both are Avengers). Other skolems, such as α -lawyer or β -detective, exploit the taxonomic category field of the NOC

list. In such cases, the most similar member of the category is chosen to resolve the skolem, so α -lawyer is filled with a character similar to α that is also a lawyer, and β -detective is filled by a detective that resembles β (e.g., Miss Marple for Stephen Hawking). No skolem is ever instantiated as a character that is already in use in the current story context.

These additions to *Scéalextric* give it much of the flexibility of traditional story grammars while preserving the key knowledge structures that make its stories so playful and diverse. Its stories still exploit unexpected juxtapositions of NOC characters that evoke both similarity and incongruity, but now a story can draw even more characters into its web while choreographing how they interact with each other. As we consider this an important contribution of the paper we shall make these additional triples and skolemizations available for use by other story-generation researchers. But now let us consider how these additions can be exploited at the meta-level to drive a creative story-telling double act.

Are These The Droids You're Looking For?

In comedy, timing is key, and so choreography is needed to align the actions of partners to ensure that they read from the same script while staying in sync from one beat to the next. For a given beat it is impractical for one to infer the timing of another, as a *NAO* cannot reliably infer how long it will take *Alexa* to speak the text of a beat, just as *Alexa* cannot know how long the *NAO* may take to enact it. If our duo is not to become hopelessly co-dependent, an unseen partner is required to manage backstage coordination. This 'third man' is a *blackboard* (Hayes-Roth, 1985), the ideal architecture for synchronizing the cooperative strangers of a distributed system. As shown in Montfort *et al.* (2013), a blackboard is a communal scratch pad on which different generators can track their work and share both knowledge and intermediate work-products. We shall use a blackboard to store key elements of the domain- and meta-models of the performers, as well as their current positions in each.

The double-act is initiated by a command to *Alexa*, such as 'Alexa, tell me a story about Donald Trump.' So it is the responsibility of *Alexa* to retrieve an apt tale from her story space, as already pre-generated using the augmentations to *Scéalextric* described above. Each story is fully rendered as text when retrieved, and *Alexa* segments it into a sequence of individual story beats of one action apiece. It is this sequence that is placed on the blackboard for *NAO* to see. In the dance of *Alexa* & *NAO*, *Alexa* leads and *NAO* follows. *Alexa* starts the tale by articulating the text of the first beat, then waits for *NAO* to respond. The robot, seeing the cued beat on the blackboard, reacts appropriately, either with a pantomime action for the plot verb, or with a gesture that signifies its response to the story so far. But *Alexa* does not proceed with the story until she is given an explicit vocal command to do so, e.g., 'continue', 'go on', 'then what' or 'tell me more.' This can come from the audience, but *NAO* will provide it itself if none is forthcoming. When it replies to *Alexa*, the robot looks down at the *Echo* device to maintain the social contact of a double-act. Both agents are engaged in a back-and-forth conversation, and it should show.

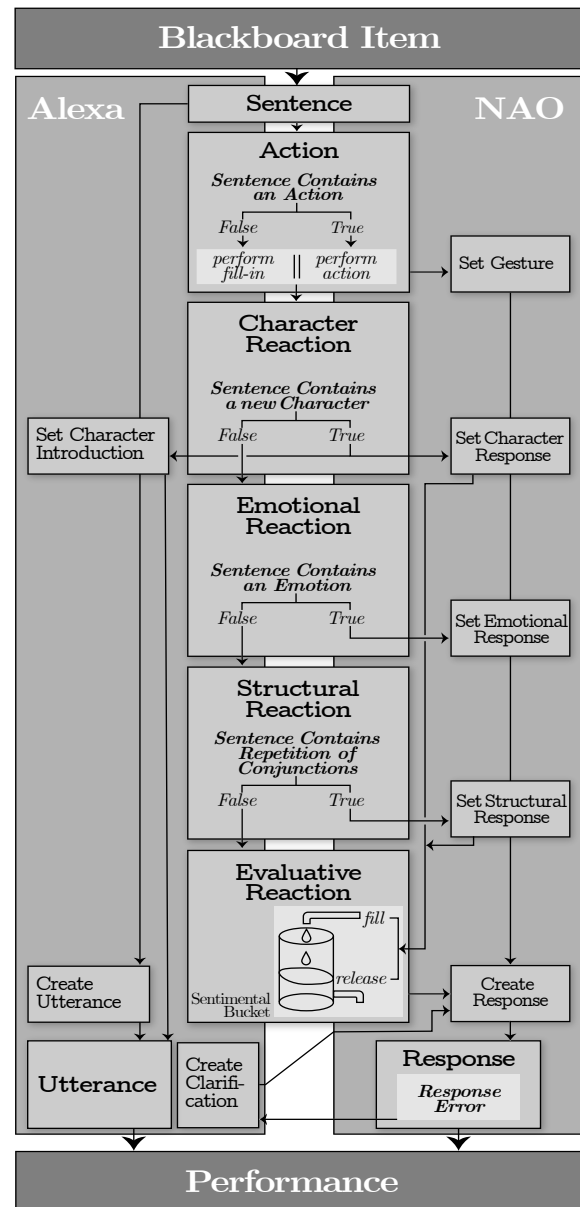


Figure 2. Blackboard logic for the system's meta-models.

This baseline conversation uses only the domain models. But as more substance is added to the meta-models of each partner, sophisticated artifice is possible. So *NAO* can peek at the next story beat on the blackboard, and determine its causal relation to the last. It can then use this to choose its cue to *Alexa* to proceed with the tale. Suppose the next beat is 'But Donald spurned Hillary's advances'. Seeing the *but*, *NAO* can prompt *Alexa* to go on by ominously asking 'But then what?' In this way a single initiative task becomes a mixed initiative task, in which *NAO* draws the tale out of its companion, and seems to shape it as it is spun. As *NAO* uses pre-recorded sound cues for these interactions (recall that *Alexa* does not understand *NAO*'s native voice), it can

use sound effects here as well as richly tempered voice recordings, to give the interactions a greater social dimension.

An integrated depiction of the double-act's meta-models is shown in Figure 2. A key responsibility of a meta-model is to predict an audience's response to an unfolding story and allow performers to take elaborative action as needed. Suppose *Alexa* articulates three successive story beats that begin with *then*, *so*, or *and*. A meta-model may see this as characteristic of a flat stretch in a story in which one action leads predictably, and boringly, to the next, and so spur the robot to reply with a structural reaction, such as a yawn.

If *NAO* peeks ahead to see that the current flat stretch is about to lead to a 'but' it can announce, wisely, 'I see a but coming.' Alternatively, the robot might reply with laughter when a silly act is described, or, more insightfully, when a character gets his comeuppance. An unexpectedly negative turn in a story may prompt the robot to utter "Dick move!" or some other pejorative that shapes the audience's view of the evolving tale. The robot can also pass remarks on characters as they are introduced into the story, by querying the NOC list for relevant qualities. So it may, for instance, say that "Donald Trump is so arrogant" when that character is introduced for the first time. Each meta-model may also be capable of its own small acts of creativity. For instance, the meta-model can generate dynamic epithets for characters as they evolve in a tale, such as *Hillary the Death-bringer*, *Bill the Seducer*, or *Donald the Lie-Teller*. These epithets can be the robot's spoken contribution to the plot delivery. So the meta-model allows performers to switch from narrator to actor to Greek chorus as the story context demands.

The joint meta-model of Fig. 2 supports the following reactions to a tale as it is told: *gestural reactions* (the *NAO* makes an appropriate gesture for a given action); *character reactions* (*NAO* or *Alexa* react in an apt fashion whenever a character is introduced); *structural reactions* (*NAO* reacts to the logical shape of the tale); *emotional reactions* (*NAO* reacts with emotion to a plot turn that is highly positive or negative); and *evaluative reactions* (*NAO* or *Alexa* react to their cumulative impression of a story so far, if this opinion is sufficiently positive or negative to be worthy of remark). Since our content model is *Scéalextric*, a wholly symbolic CC system, all stories have predictable markers that allow our meta-models to be implemented as rule-based systems. The next section illustrates the reaction of the meta-models within an annotated transcript of our double act in action.

The Double Act in Action

As the only embodied agent in the duo, it is the responsibility of the robot to create the duo's shared physical space. *NAO* must address itself to *Alexa* to present their interactions as a conversation, and not just a pairing of devices that speak past each other in a synchronized manner. To begin with, *NAO* asks *Alexa* to 'open your notebook' so they can create a story together. *Alexa* then asks *NAO* for a subject, which it provides (such as "Star Wars") and *Alexa* reacts by noting her satisfaction with the story to come. This tale then unfolds, beat by beat, with *NAO* asking *Alexa* to continue between beats once it has enacted its own reply. But

Alexa has responsibilities too, and must do in words what it cannot do with physical acts. *Alexa* must acknowledge the robot's contributions to show that they do indeed share the same space. For when one agent acts as an audience to the other, they can collectively shape *our* feelings for the tale.

Alexa's weaknesses have been well-documented in other work (e.g., Kapadia *et al.*, 2017), and her recovery mode is not sufficiently transparent to avoid failed interactions with the *NAO*. So if *Alexa* does not receive her next prompt in a timely manner, she will reiterate several requests for input before eventually quitting the narrator skill. Unfortunately, this reiteration cannot be unpacked so as to tell the blackboard of a failed interaction, so *NAO* will remain unaware that *Alexa* has, so to speak, left the building. However, the robot can detect a long pause in a conversation and prompt *Alexa*, with comic frustration, to continue. With each of its reprompts, the robot's feigned exasperation grows, until it finally tells *Alexa* to "reopen your notebook" and continue. At this point, the narration continues from the last beat that was processed, and the duo returns to their bickering ways.

What follows is a sample transcript of the double-act in action. All speech is presented in italics, while each gesture and meta-model reaction is annotated in square brackets:

NAO: Alexa, open your notebook.
Alexa: Please provide a topic for our story.
NAO: How about Harry Potter?
Alexa: I have written a great story about Harry Potter.
NAO: Let's hear it.
Alexa: What if Harry fell in love with Hermione Granger?
NAO: Sounds promising.
Alexa: At first, Harry's thoughts of Hermione were of love.
NAO: Proceed. [Love Gesture]
Alexa: I didn't get that. [Failed Interaction]
NAO: Please go on. [Reprompt] [Waiting Gesture]
Alexa: Because Hermione openly favored Harry.
NAO: Tell me more.
Alexa: Hey, Bubblehead! we need another character.
[Introduction of new character]
NAO: Lizzy Bennet works for me. [New character Reaction]
Alexa: Yet Hermione governed the land with Lizzy Bennet.
NAO: Oh damn. [Structural Reaction]
Alexa: Should I continue? [Reprompt]
NAO: Continue. Is there more? [Waiting Gesture]
Alexa: But Harry spurned Lizzy Bennet's advances.
NAO: Oh my goodness! [Structural Reaction]
Alexa: So Lizzy Bennet took a whip to Harry's back.
NAO: Please go on.
Alexa: Well, Harry complained to Hermione.
NAO: But then what? [Structural prediction]
Alexa: But Lizzy Bennet spread slander about Harry.
NAO: That is interesting. [Spread Gesture]
Alexa: Yet Hermione's suspicions about Lizzy grew.
NAO: That escalated quickly! [Structural Reaction]
Alexa: Are you still there? [Reprompt after brief failure]
NAO: Tell me more. [Waiting Gesture]
Alexa: Then Hermione banished Lizzy from the kingdom.
NAO: I am entertained. [Evaluative Reaction]
Alexa: But Lizzy was not as dumb as Hermione imagined.

NAO: *Oh my goodness!* [Structural Reaction]
 Alexa: *So Lizzy unseated Hermione from her perch.*
 NAO: *Then what?* [Flat Structural Reaction]
 Alexa: *Then Lizzy Bennet had Hermione locked up.*
 NAO: *So then what?* [Locked-Up Gesture]
 Alexa: *So Harry's denunciation of Lizzy was scathing.*
 NAO: *Please go on.* [Humiliated Gesture]
 Alexa: *Well, Lizzy turned Harry into an indentured slave.*
 NAO: *What a terrible person.* [Emotional Reaction]
 Alexa: *But Harry escaped from under Lizzy Bennet's nose.*
 NAO: *I am surprised!* [Structural Reaction]
 Alexa: *Now expect to be poisoned with bad cooking!*
 [New Character Introduction: Bridget Jones]
 NAO: *Bridget Jones, a great add!* [Character Reaction]
 Alexa: *Meanwhile Bridget guarded Hermione closely.*
 NAO: *Cool, tell me more.* [Guarding Gesture]
 Alexa: *So Harry's wiles proved too strong for Bridget.*
 NAO: *And then what next?* [Flat Structural Reaction]
 Alexa: *Then Harry put poison in Bridget Jones's cup.*
 NAO: *That took a dark turn.* [Emotional Reaction]
 Alexa: *After this, Harry gave Hermione her freedom.*
 NAO: *I love to hear that.* [Emotional Reaction]
 Alexa: *So Hermione fomented rebellion against Lizzy.*
 NAO: *And then what?* [Rebellious Gesture]
 Alexa: [Interaction failure. Alexa exits story skill.]
 NAO: *Continue.* [Reprompts after delay]
 NAO: *Are you there?* [Reprompts Alexa again]
 NAO: *Hello, Anyone home?* [Emotional Reaction to Alexa]
 NAO: *Alexa, reopen your notebook.* [Reopens skill]
 Alexa: *Then Hermione usurped all of Lizzy's power.*
 NAO: *So what?* [Grasping Gesture]
 Alexa: *So Hermione turned Lizzy into an indentured slave.*
 NAO: *That took another dark turn.* [Locked-Up Gesture]
 Alexa: *In the end, Harry walked Hermione down the aisle.*
 NAO: *That's just lovely.* [Emotional Reaction]
 Alexa: *The end.*

A single transcript can be revealing about specifics, but not about the generic tendencies of a system. To appreciate the latter, we simulated the double-act 136,787 times, choosing a different story to tell each time. Our goal was to estimate the relative occurrence of alternate meta-model reactions to the story in each case. In particular, we considered the following: the BUT structural reaction to a turn in the plot; the BORED evaluative reaction to a predictable stretch of plot; the STRONG emotional reaction to a highly-charged plot verb; the GOOD evaluative reaction to an exciting stretch; the NEW character reaction to the introduction of another named entity to a story; and the GESTURE reaction, which delivers a mimetic response to a given plot action. Overall, the BORED evaluative reaction accounts for 18.4% of all reactions, the BUT structural reaction accounts for 16.6%, the STRONG emotional reaction accounts for 15.5%, the NEW character reaction accounts for 7.7%, and the GOOD evaluative reaction accounts for 4%. In all remaining cases, or 37.8% of the time, the NAO responds structurally, with a prompt to “continue” or “go on” and a downward glance at the *Echo* unit by its side. The GESTURE reaction is independent of these other reaction types, since the robot can

make a gesture *and* utter a spoken response in a single turn. For 49.6% of story beats the robot performs a gesture that is visually mimetic of the current plot verb; for the other 50.4% of beats, *NAO* makes a ‘holding’ gesture – such as folding its arms, putting its hands on its hips, or shifting its weight from one leg to another – in the manner of human listeners who wish to emphasize their physical presence.

Related Work

The *Alexa* skill store contains an array of storytelling skills for the Amazon *Echo*, ranging from linear narratives to the *choose-your-own-adventure* style of story. None, however, uses computer-generated tales as a basis for narration, and few tell stories as complex or data-rich as those used here.

Kapadia *et al.* (2017) paired *Alexa* to *YuMi*, a two-armed industrial robot, to develop a learning-from-demonstration (or LfD) system. LfD requires trainers to use both hands to move a robot’s own limbs into the poses it must learn, and to annotate these actions at the same time. The pairing with *Alexa* allows trainers to speak to the LfD system to verbally label what is being taught as they use their own hands to move the robot into its demonstration poses. The authors note the vexing technical challenges that *Alexa* entails, but still argue that using *Alexa* for hands-free vocal control in a robotic context is worthwhile. Their LfD system, *EchoBot*, is not a true double-act, however, as a human manipulates both devices simultaneously with voice and gesture inputs, and *EchoBot* is not designed to exhibit its own personality.

Fischer *et al.* (2016) also use *Alexa* as voice control for a robot, the one-armed *Kinova Jaco*. Users issue commands to *Alexa* (via *Echo*) and a backend turns these commands into appropriate kinematics for the robot. While *Alexa* and the robot are cooperating partners, interaction is one-way and not a dialogue. Neither is it part of a creative task.

Kopp, Bergmann & Wachsmuth (2008), building on the work of Kita and Özyürek (2003), presented a multi-modal system that also uses a blackboard to integrate spoken text and embodied gestures into a single communicative act. In this case, multimodality occurs within the simulated environment of a virtual visual agent, or avatar, whose animated gestures achieve both communicative and cognitive ends: they augment what is said, and reveal the inner state of the cognitive agent as they do so. Each modality operates with a shared representation on the blackboard (both imagistic and propositional in nature) of that which is to be said, and enacts it as speech or gestures to suit their own agendas. In effect, this system is a double act of sorts, realized as just a single coherent agent. Yet such coherence prohibits a dual system from reaping the benefits of a true double act, since only the latter allows a system to talk to, interrogate, and make fun of itself in a consistent and humorous manner.

Farnia & Karima (2019) explore how humorous intent is marked in a text, and the effect of these markers, subtle or otherwise, on the perception of humour by an audience. A double act of *Alexa* and *NAO* allows us to explore markers that are more than just textual, or even vocal, to explore how a witty personality can be constructed from the physical and meta-linguistic markers that are imposed on a text.

Double Vision: Summary and Conclusions

A good double-act is a marriage of convenience, even if it often looks otherwise. Many comedic duos go out of their way to accentuate their differences, as comic friction only serves to emphasize their complementarity. When partners complete each other, it is as if they occupy a world all their own. Nonetheless, even a seamless partnership may require significant backstage coordination to make it all work. The same is true of technology double acts, such as our pairing of *Alexa* & *NAO* that turns story-telling into a performance. In this paper we have focused on the considerable – but not always obvious – technical challenges of making a double act of *Alexa* and *NAO* a practical reality in a CC context. We have developed the content models, the meta-models, and the platform functionalities to the point where we can finally use the double act to empirically test our hypotheses regarding the true value of embodiment and multimodality in the generation and delivery of machine-crafted artifacts.

Our double act divorces the job of story generation from the task of telling a story well. Each responsibility requires one CC system to be sympatico with the other, just as the performers in a double act must read each other's minds, or – more realistically – their shared blackboard architecture. Nonetheless, we have structured the performative functions so that they can work with machine-generated tales of any kind, once the meta-models have been adapted to operate over this new content model. Even so, we have only begun to exploit the full performance possibilities of offline generation and later online delivery in a multimodal setting. In addition to the obvious entertainment applications, we are mindful of the educational possibilities of CC double acts that *show* as well as *tell*, that embody what they create, and that reveal an emergent personality they can call their own.

To both see and hear the *Walkie Talkie* double-act do this thing, readers are invited to subscribe to the following channel on *Youtube*, where annotated videos of the duo performing a series of different stories can be watched online:

<https://bit.ly/2SNeeHQ>

References

- Amazon. (2019). Alexa skills kit. <https://developer.amazon.com/alexa-skills-kit> (last accessed, February 2019).
- Bergson, H. (1911/2013). *Laughter: An Essay on the Meaning of the Comic*. Trans. C. Brereton & F. Rothwell. New York: Dover.
- Epstein, L. (2005). *Mixed Nuts: America's love affair with comedy teams from Burns and Allen to Belushi and Aykroyd*. Waterville, ME: Thorndike Press.
- Farnia, M. & Karimi, K. (2019). Humor markers in computer-mediated communication. Emotion perception and response. *J. of Teaching English with Technology*, 1:21:35.
- Fischer, M, Memon, S. & Khatib, O. (2016). From Bot to Bot: Using a Chat Bot to Synthesize Robot Motion. *The AAAI Fall Symposia series, AI for Human Robot Interaction*, TR FS-16-01.
- Freud, S. (1919). Das Unheimliche. In *Collected Papers*, volume XII. G.W. 229–268.
- Fuller, D. & Magerko, B. (2010). Shared mental models in improvisational performance. In Proc. of the Intelligent Narrative Technologies III Workshop (INT3 '10). ACM, New York, NY, USA
- Hayes-Roth. B. (1985). A Blackboard Architecture for Control. *Artificial Intelligence*. 26 (3): 251–321.
- Kapadia, R., Staszak, S., Jian, L. & Goldberg, K. (2017). EchoBot: Facilitating Data Collection for Robot Learning with the Amazon Echo. In Proc. of the 13th IEEE Conference on Automation Science and Engineering (CASE) Xi'an, China, August 20-23.
- Kita, S., & Özyürek, A. (2003). What does cross-linguistic variation in semantic coordination of speech and gesture reveal?: Evidence for an interface representation of spatial thinking and speaking. *Journal of Memory and language* 48(1):16–32.
- Kopp, S., Bergmann, K., & Wachsmuth, I. (2008). Multi-modal communication from multimodal thinking towards an integrated model of speech and gesture production. *International Journal of Semantic Computing* 2(1):115–136.
- Pelachaud, C., Gelin, R., Martin, J., & Le, Q.A. (2010). Expressive gestures displayed by a humanoid robot during a storytelling application. *New Frontiers in Human-Robot Interaction (AISB)*, Leicester, UK.
- Pérez y Pérez, R. (2007). Employing emotions to drive plot generation in a computer-based storyteller. *Cognitive Systems Research* 8(2):89-109.
- Montfort, N. (2009). Curveship: An interactive fiction system for interactive narrating. In Proc. of the Workshop on Computational Approaches to Linguistic Creativity, at the 47th Ann. Conf. of the Assoc. for Computational Linguistics, Boulder, Colorado, 55–62.
- Montfort, N., Pérez y Pérez, R., Harrell, F. & Campana, A. (2013). Slant: A blackboard system to generate plot, figuration, and narrative discourse aspects of stories. In Proc. of the 4th International Conf. on Computational Creativity. Sidney, Australia, June 12-14.
- Shaftesbury, Lord, (1709/2001). Sensus Communis: An Essay on the Freedom of Wit and Humour. In: *Characteristicks of Men, Manners, Opinions, Times*. Indiana, Indianapolis: Liberty Fund.
- Veale, T. (2016). Round Up The Usual Suspects: Knowledge-Based Metaphor Generation. In Proc. of the Meta4NLP Workshop on Metaphor at NAACL-2016, the annual meeting of the North American Assoc. for Computational Linguistics. San Diego, CA.
- Veale, T. (2017). Déjà Vu All Over Again: On the Creative Value of Familiar Elements in the Telling of Original Tales. In Proc. of ICCO 2017, the 8th Int. Conf. on Comp. Creativity, Atlanta.
- Veale, T. & Cook, M. (2018). *Twitterbots: Making Machines that Make Meaning*. Cambridge, MA: MIT Press.
- Wicke, P. & Veale, T. (2018a). Storytelling by a Show of Hands: A framework for interactive embodied storytelling in robotic agents. In Proc. of AISB'18, the Conf. on Artificial Intelligence and Simulated Behaviour, pp 49--56.
- Wicke, P. & Veale, T. (2018b). Interview with the Robot: Question-guided collaboration in a storytelling system. In Proc. of ICCO'18, the 9th Int. Conference on Computational Creativity, Salamanca, Spain, June 25-29.

Churnalist: Fictional Headline Generation for Context-appropriate Flavor Text

Judith van Stegeren
Human Media Interaction
University of Twente
Enschede, The Netherlands
j.e.vanstegeren@utwente.nl

Mariët Theune
Human Media Interaction
University of Twente
Enschede, The Netherlands
m.theune@utwente.nl

Abstract

We present Churnalist, a headline generator for creating contextually-appropriate fictional headlines that can be used as ‘flavor text’ in games. Churnalist creates new headlines from existing headlines with text modification. It extracts seed words from free text input, queries a knowledge base for related words and uses these words in the new headlines. Churnalist’s knowledge base consists of a dataset of pre-trained word embeddings, thus requiring no linguistic expertise or hand-coded models from the user.

Introduction

The field of natural language generation (NLG) investigates how texts can be created automatically. NLG systems have been used to transform data, such as weather data, football match statistics and intensive care data, into texts for a specific audience. NLG is also used for generating fictional or creative text, such as poetry (Gonçalo Oliveira 2012), lyrics (Bay, Bodily, and Ventura 2017), advertising slogans (Gatti et al. 2015), and character dialogue in games (Lukin, Ryan, and Walker 2014; Schlünder and Klabunde 2013). It is in the latter type of applications that NLG has similar research goals as computational creativity, i.e. supporting or even completely replacing a human in the execution of a creative task.

In this paper, we present *Churnalist*, an interactive system for generating newspaper headlines for a given context. Our system is meant for generating fictional headlines that can be used in games. Most headline generators take a newspaper article as input and summarize it in one sentence. In contrast to these systems, Churnalist accepts any type of text as input and generates headlines based on nouns extracted from the input text. By reusing nouns from the input text in the generated headlines, we aim to make the headlines context-appropriate, by which we mean that readers will believe that the headlines are related to the input text. We want to exploit the human tendency to see connections between texts (input text and headlines) where there are none.

There are various games that use fictional news (in the form of headlines or newspaper articles) to provide narrative context to the player. For example, in city simulation game SimCity 2000 (Maxis 1996), the player has access to newspaper articles with information about important city issues, disasters and new technologies. Similarly, Cities Skylines

(Colossal Order 2017) features a fictional social media website called ‘Chirpy’, where virtual citizens of the player’s city express their (dis)satisfaction with the player’s performance as mayor and city planner. In Deus Ex: Human Revolution (Eidos Montreal 2011), the player can find ebooks and newspapers that provide background information on the social unrest that is driving the game’s main storyline. Idle game Cookie Clicker (Thiennot 2013) has a news ticker with randomly generated headlines reflecting the player’s game progress.

The fictional newspaper articles and headlines can be seen as examples of *flavor text*, i.e. text that is not essential to the main game narrative, but creates a feeling of immersion for the player. This is especially important for role-playing games and simulation games, as it gives players the impression that the virtual world they are interacting with is a living and breathing world.

Writing flavor text is a time-consuming task for game writers. Text generation can be a solution to this problem. Most games that incorporate text generation use simple, manually created templates or canned text. More complex NLG techniques rely on linguistic models, which often take considerable effort to create and require linguistic expertise. Statistical linguistic models can be created automatically from a dataset of texts. However, generators with underlying statistical models offer less fine-grained control over the output. Canned text and simple templates offer a balance between control over the output and ease of use, but have the disadvantage that players will figure out the underlying templates after playing the same game for a while, or after replaying the game (Backus 2017). We think that NLG techniques other than canned text and simple templates are worth investigating in the context of game development, especially data-driven approaches to text generation, as these can overcome the need for expensive, handcrafted language models. We propose a system that can generate fictional headlines in order to support game writers in the task of writing flavor text.

In the next section, we discuss related work. Then, we present Churnalist and describe the system goal, the architecture and the generation steps in detail, together with an example. Finally, we discuss our results and describe some ideas for future work.

Related work

In this section, we discuss work related to headline generation, text generation for games and generative systems that take context into account.

Headline generation

Headline generation is often seen as a document summarization task, where headline generators take a full article text as input and return a headline that describes the most salient theme or the main event of the text. The literature distinguishes between extractive summarization, e.g. (Jing 2000), and abstractive summarization approaches. Contrary to extractive systems, the output of an abstractive system does not have to correspond to a sentence from the input text. Abstractive headline generation systems may be rule-based (Dorr, Zajic, and Schwartz 2003), statistics-based (Banko, Mittal, and Witbrock 2000) or based on machine learning (Colmenares et al. 2015; Shen et al. 2017), with the latter winning in popularity in recent years.

Headlines (Gatti et al. 2016) is an example of a headline generation system that focuses on the creative side of writing headlines. It can be used to support editors in their task of writing catchy news headlines. Given a newspaper article text as input, it extracts the most important words from the text and uses these as seed words for generating a large set of variations on well-known lines, such as movie names and song lyrics. This research is a good example of combining NLG with techniques from computational creativity.

Text generation for games

Text generation for games is a form of *procedural content generation* (PCG). Procedural content generation, which refers to the creation of content automatically through algorithmic means, is a relatively new addition to the field of artificial intelligence. PCG for games studies the algorithmic creation of *game contents*, defined by Yannakakis and Togelius (2011) as all aspects of a game that affect gameplay but are not non-player character behavior or the game engine itself, such as maps, levels, dialogues, quests, music, objects and characters. Text generation techniques can be used for generating dialogue, stories, quests and flavor text for games. Although including generated game text in video games is winning in popularity, these texts are often generated with simple NLG techniques, such as canned text and simple templates.

On the other hand, within the natural language generation field, there are various publications that list game text as a possible application (Schl nder and Klabunde 2013; Strong et al. 2007; Lukin, Ryan, and Walker 2014). However, there are few cases where the implemented system is actively used in a games context. One example is Caves of Qud (Freehold Games 2018), which combines techniques from PCG and NLG to create a unique game world for every play-through. The developers of Caves of Qud used a hand-written knowledge base for their text generator, which links in-game themes to a set of words and phrases (see next section). In our research, we use a knowledge base for a similar purpose: to link seed words from the input text to a set of related words. Instead of creating it manually, we used word embeddings as the basis for our knowledge base.

Generative systems and context

For Churnalist, we were inspired by how other generative systems create text for a given context. Context is a slippery notion. Within PCG for games, generated artifacts are judged together with the rest of the assets of the game for which they were generated. In the case of Churnalist, context means the input text and, more generally, the game from which the input text is taken and for which the headlines are generated.

In slogan generation it is also important for the generated texts to fit a given context or domain. In BRAINSUP ( zbal, Pighin, and Strapparava 2013), the generated slogans must fit a domain for which the user manually supplies keywords as input to the system. In BISON (Repar et al. 2018), keywords are automatically extracted from two sets of documents representing two domains that the generated slogans must fit. The pool of extracted keywords is expanded using FastText embeddings (Bojanowski et al. 2017). The keywords are then used to fill the slots in templates (‘slogan skeletons’) derived from a corpus of slogans. The BISON approach to extracting and expanding the set of keywords is very similar to that of Churnalist, as we will see in the following sections. One way in which Churnalist differs from both BRAINSUP and BISON is that those systems derive syntactic patterns or templates from a corpus and then fill their slots, whereas Churnalist takes the original texts from a corpus and applies word substitution to them. In that respect, Churnalist is more similar to the transformation-based approach to lyrics generation proposed by Bay, Bodily and Ventura (2017).

Another system related to Churnalist is O Poeta Artificial 2.0 (Gonalo Oliveira 2017), a bot tweeting poems that are generated for trending hashtags on Twitter. It uses hashtags as topical seed words to generate poems that fit the hashtag. The bot is based on PoeTryMe (Gonalo Oliveira 2012), a poem generation framework for Portuguese, which uses external data sources to enrich its output, such as a database of Portuguese poems, a semantic graph and lexical datasets. Churnalist has multiple things in common with O Poeta Artificial: both generate text for a specific context, work with seed words and external semantic resources, and need a method to deal with out-of-vocabulary words in the input.

Caves of Qud’s text generation (Grinblat and Bucklew 2017) influenced our design for Churnalist as well. The game generates fictional biographies for mythical non-player characters called sultans. These biographies consist of randomly generated fictional events from the life of the sultan, such as starting a war, acquiring a mythical weapon or forging an alliance. To infuse a sense of coherence in these biographies a domain, such as ‘glass’, ‘jewels’, ‘ice’ or ‘scholarship’ is assigned to each sultan. To tie the life events in the biography together, the generator incorporates domain-specific elements in each event.

Players of Caves of Qud will interpret the randomly generated biographies as coherent narratives, thereby creating their own logical explanation for the overarching theme in each biography. The developers call this human tendency to perceive patterns ‘apophenia’. It is related to the ‘charity of interpretation’ effect studied by Veale (2016), who found that “readers will generously infer the presence of meaning in texts that are well-formed and seemingly the product of

an intelligent entity, even if this entity is not intelligent and the meaning not intentional.” If humans see a text in a well-known form (or *container*), they are disposed to attribute more meaning to the text than it actually contains. A similar effect is the Eliza effect described by Hofstadter (1995), who noticed that humans will attribute intelligence or empathy to (text-producing) computer systems. This approach to evoking context is also related to the ‘intention’ aspect of framing information (Charnley, Pease, and Colton 2012) in computational creativity. With Churnalist, we want to exploit this effect too: by incorporating words from the input text in the output, we hope that readers will perceive the generated headlines as coherent with the input.

Description of Churnalist

Churnalist is a system for generating fictional headlines that are context-appropriate for the textual input. In this section, we discuss the goal of the system and the requirements for the output. We elaborate on the technical design of the system and provide a running example.

System goal

Game writers can use Churnalist for generating flavor text for video games, in the form of headlines. Instead of taking newspaper article texts as input, as is common practice for headline generators, Churnalist accepts user-supplied free text as input, in the form of English sentences from a game. For example, see the one-sentence input in Figure 1.

“Mario must save Princess Peach from Bowser’s castle.”

Figure 1: An example of valid input text. The names and noun phrases that Churnalist will incorporate in the output headlines are underlined.

Churnalist extracts a set of seed words from the input and creates new headlines by doing word-substitution on headlines from a database. The seed words consist of words from the input. We expand the set of seed words by querying a vector space of word embeddings for vectors close to the words from the input. By using words that have a link with the input text, or *context words*, we generate headlines that fit the context that is represented by the input. By inserting context words in the headlines from the database, we hope to exploit the Eliza effect (Hofstadter 1995), apophenia (Grinblat and Bucklew 2017) and the charity of interpretation (Veale 2016) in readers: readers should think that the headlines are related to the context. Churnalist’s output is a set of fictional headlines, like in Figure 2. A more extensive example of using game text as input is provided in Figure 4.

Using free text input makes Churnalist usable for different games and different topics. Regardless of the content or the type of game, as long as the input text contains content words (nouns), Churnalist will extract these from the input and use them as seed words to generate headlines. Churnalist was developed using publicly available datasets, open source libraries and only simple text modification techniques, so that for future users no linguistic expertise is required.

Mario apologises to mother involved in car crash
 Mario injured after Sicily volcano triggers earthquake
 Mario says Arsenal return vs Qarabag was ‘emotional’
 Mario: ‘My marriage is over because I voted to leave the EU’
 Princess Peach unveils world’s first Chromebook with AMD processors
 Bowser’s castle retains Border-Gavaskar trophy after cleaning up Australia on day five

Figure 2: Example output text for Churnalist, given the input text in Figure 1.

For Churnalist’s output, we adopt similar requirements as Gonçalo Oliveira (2012):

1. The output texts must look like headlines. We are not generating news article texts. The content of the headlines does not have to be realistic or ground in reality. On the contrary: we aim for fictional output, as well as output that is not literally present in the database of headlines (for copyright reasons).
2. Headlines must be grammatical.
3. Headlines must feel context-appropriate (coherent, meaningful, relevant) for the input text to a not-too-discerning, not overly critical reader.

Architecture

We have implemented a prototype system with the architecture shown in Figure 3. Churnalist has a modular pipeline design so that every subtask can be implemented according to the requirements of the user, to make the system as flexible as possible. The pipeline consists of three modules, one for each step in the generation process. At the end of each step, the user of Churnalist can manually filter the output of the system, thus fine-tuning the nouns and noun phrases that are used in later generation steps.

The first module, the *keyword extractor*, reads the input text and extracts the most important words. These words are the *seed words*. The second module takes the list of seed words and expands this with a set of loosely related words, gathered from the *knowledge base*. The seed words and the related words form the set of *context words*. The *substitution module* takes a random headline from a headline database, runs it through a dependency parser and substitutes parts of the sentence with context words. The resulting new headline is the output of the system. Users can generate multiple headlines from one input text; the number of possible results is determined by (1) the number of seed words in the input text, (2) the size of the headline database and (3) the size of the set of user-approved context words.

In the rest of this section, we describe these steps in more detail and provide an example. Figure 4 shows an input text taken from a dilemma-driven serious game. It features both a situational description and some lines of NPC text. The rest of this paper will feature examples that were generated with this input text.

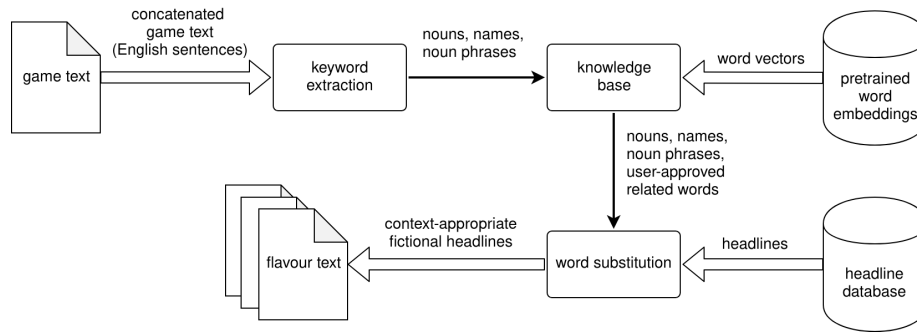


Figure 3: Churnalist: system architecture

“You are the system administrator of SuperSecure Ltd, a hosting company. At four o’clock in the afternoon, your manager storms in. Apparently, there has been a break-in in your computer network. The CEO has been receiving anonymous emails from a hacker that demands a payment of \$100,000 before midnight. If SuperSecure does not pay, they threaten to publish sensitive company documents online. The manager is worried, since the hacker claims to possess important intellectual property. Manager: Can you find out how the hackers got into our systems? CSIRT: We recognize this mode of operation. We will share some relevant IOCs with your company. Can you contact us if you have finished your forensical analysis? Security officer: There has been a nation-wide increase in phishing attacks in the past few days. System administrator: I can’t find any traces of active malware on our Windows server. I will check the network log files for malicious activity.”

Figure 4: Representative input text for Churnalist: game text from a dilemma-based serious game, consisting of a description and a few lines of NPC text. Names and noun phrases are underlined.

Keyword extractor

The start of the pipeline is the keyword extractor. We assume that the input text consists of grammatical sentences, so that it can be parsed by a sentence tokenizer and a dependency parser. The keyword extractor runs the input text through the NLTK sentence tokenizer¹ and the spaCy² part-of-speech-tagger and dependency parser trained on spaCy’s default corpus for English.³ It uses spaCy’s noun phrase extraction to extract all English noun phrases from the input text. The keyword extractor saves all noun phrases that occur in the input text, together with the head of each noun phrase. Churnalist uses the head nouns as seed words and saves the noun phrase itself so it can be reused later, during the word substitution phase. See Figure 5 for an example of seed words and the corresponding noun phrases as extracted from the input text in Figure 4.

Knowledge base

In order to get more variety in our output, and not limit the words used in our output to nouns extracted from the input text, we extend the list of seed words with related words. Note that we mean ‘related’ in a broad sense; not just synonyms. To obtain these words, Churnalist queries the knowledge base for words similar to the seed words. We took this idea from the procedurally generated biographies in Caves of Qud (Grinblat and Bucklew 2017), which evoked a feeling of coherence because of the related domain words that were put into the biography. For example, for the seed word

‘ice’ the words ‘lightblue’, ‘frost’, ‘cold’ or ‘winter’ would all be suitable related words. The word lists for the various domains in Caves of Qud were written manually by the game developers. However, we do not want to build large content models by hand. Instead, we want to focus on generating text from data that can be obtained automatically. Like Repar et al. (2018), we used the English dataset of FastText’s pre-trained word embeddings (Bojanowski et al. 2017) as a knowledge base, similar to the external semantic datasets used by the PoeTryMe framework (Gonçalo Oliveira 2012).

Word embeddings are a method for encoding words as their context, based on a corpus. The FastText dataset is based on a corpus of Wikipedia articles. It contains words represented as vectors that encode the context of these words. Words (vectors) that are close to each other in the resulting vector space, are words that occur in similar contexts.

A useful property of the FastText dataset is that it contains word embeddings that encode subword information: the vector of a word is created from the vectors of its subwords of length n . As a consequence, the dataset can be used to obtain vectors for out-of-vocabulary words: we only need to create a vector for them by looking at the vectors of their subwords. This allows us to deal with words that are not present in the semantic resources being used. Consequently, we bypass a problem similar to O Poeta Artificial’s out-of-vocabulary hashtags (Gonçalo Oliveira 2017). Another advantage of using FastText is that its datasets are available in multiple languages, which allows us to port our system to languages other than English (for instance, Dutch).

The seed words are passed on to the knowledge base, which tries to assign a vector to each word and find its closest neighbours. If the seed word is an out-of-vocabulary word,

¹NLTK 3.3, <https://www.nltk.org>

²spaCy 2.0.16, <https://www.spacy.io>

³Language model en_core_web_sm 2.0.0

Head noun	noun phrase
administrator	system administrator
company	hosting company, company
network	computer network
CEO	CEO
emails	anonymous emails
documents	sensitive company documents
manager	manager
property	important intellectual property
hackers	hackers
IOCs	relevant IOCs
analysis	forensical analysis
officer	Security officer
traces	traces
malware	active malware
server	Windows server
files	network log files
activity	malicious activity

Figure 5: Noun phrases and their head noun that the keyword extractor extracted from the input text from Figure 4. Generic phrases, such as ‘days’, ‘o’clock’ and ‘afternoon’, were removed manually from the list of seed words.

Word	distance	remark
companiess	0.7817	typographical error
subsidiary	0.6847	
telecompany	0.6821	too specific
companywide	0.6773	not a noun
ecompany	0.6668	too specific
webcompany	0.6496	
corporation	0.6315	
firm	0.6218	

Figure 6: Examples of suggestions from the knowledge base for the word ‘company’, together with the distance between the word vector for ‘company’ and the word vector for the suggestion. The knowledge base lists results in descending order of distance to the seed word. The final column lists reasons for rejecting this word. Not all suggestions by the knowledge base are shown.

the system calculates a new vector for the word based on the word embeddings of its subwords, and uses this new vector to find related words. The user can set a minimum distance for suggestions from the knowledge base and select which suggested words should be passed on to the substitution step. For an example of the results of the knowledge base, see Figure 6.

The knowledge base contains a machine-learned word embeddings model that was trained on Wikipedia dumps. Consequently, there are words in the model that are unsuitable for inclusion in Churnalist’s output, such as words with crowd-sourced typographical errors. For example, the words closest to ‘company’ are ‘companiess’, ‘companythe’ and ‘companyx’, which result from typographical errors (and possibly pre-processing errors) in the Wikipedia dataset. Additionally, some words are very similar to one of the seed words but have no connection to the way that seed word is used in the input text. Take the compound noun ‘security officer’, which

means someone who defines and enforces the information security policy in a company. Its head noun is ‘officer’, for which the KB will list ‘sergeant’, ‘quartermaster’ and ‘sub-lieutenant’ as related words. However, these words have little connection with the term ‘security officer’ and should not be used in the output. The user of Churnalist can filter such unsuitable suggestions for related words from the knowledge base.

The set of seed words together with the set of related words from the knowledge base forms the set of *context words*. Figure 7 shows the final set of context words for the seed words from Figure 5.

Substitution module

The substitution module receives the list of context words from the knowledge base module and produces new headlines that contain one or more context words. It creates new headlines by substituting the subject of an existing headline from the headline database. This approach is similar to that of Headlines (Gatti et al. 2016), which inserts keywords from a newspaper article into existing sentences.

As external dataset of headlines, we use a collection of headlines scraped with the API from News API.⁴ This API returns headlines and article excerpts from several large news websites. We collected 3629 headlines from media from the UK and the US in December 2018 and January 2019.

The substitution module starts by picking a random headline from the headline database. This headline is used as the starting point for one new headline. The headline is run through spaCy’s part-of-speech-tagger and dependency parser, trained on spaCy’s default English corpus. From the information of the parser, Churnalist tries to find the subject of the sentence. This is the substitution target. If the parser cannot determine what the subject of the sentence is, a different headline is drawn randomly from the headline database.

Next, Churnalist chooses a random seed word. Each seed word has a set of context words associated with it: the seed word itself, noun phrases from the input, and the user-approved related words from the knowledge base. Churnalist randomly chooses one of these as a substitution candidate. If the substitution target is of a different number than the substitution candidate, Churnalist converts the candidate to the right number (singular to plural or vice versa). Finally, the target is substituted by the candidate and the new headline is presented to the user.

Results

In this section, we discuss our results. Figure 8 shows examples of generated headlines, together with the original headline and seed word. Consider the requirements we mentioned earlier: generated headlines should have an appropriate form, should be grammatical and should be context-appropriate for the input text.

Firstly, applying text modification to the headlines will lead to texts that again look like headlines. Informal inspection of the headlines generated suggests that this requirement is

⁴News API, <https://newsapi.org>

Seed word	approved suggestions	rejected suggestions
administrator	-	administratorship, administrator, nonadministrator
company	subsidiary, webcompany	companiess, companythe, companynew
CEO	executive, shareholder, entrepreneur, investor	CFO, COO, CTO
network	-	networky, networkx, networknbc
emails	-	emailings, voicemails, emailers
documents	documentation, memos, archives	documentations, documen, documentries
manager	teammanager	managership, imanager, managerin
property	-	poperty, propert, propertyless
hackers	hacktivists, cybercriminals, scammers	hackings, blizzhackers, hackery
IOCs	-	ligtvoet, zeijst, lennaert
analysis	-	analyses, analysises, analysisist
officer	-	underofficer, officerer, commander
malware	spamware, botnet, vulnerabilities	spyware, malwarebytes, antivirus
server	-	iserver, vserver, pvserver
files	folders, fileserver	fileset, filesmy, filespace
activity	-	activityi, activism, activin, reactivities

Figure 7: Seed words and examples of knowledge base suggestions for related words. Not all words suggested by the knowledge base are shown. The results were approved and rejected manually by the first author. Words in the ‘approved’ column are added to the set of context words.

Seedword	system administrator
Headline	Revealed: 500k number plate conman is a convicted people smuggler
Output	Revealed: system administrator is a convicted people smuggler
Seedword	hosting company
Headline	Pelosi has edge over Trump on budget negotiations, CBS News poll shows
Output	Hosting company has edge over Trump on budget negotiations, CBS News poll shows
Seedword	computer network
Headline	Met Office issues ice warning as snow hits UK
Output	Computer network issues ice warning as snow hits UK
Seedword	hacker
Headline	Uber loses latest legal bid over driver rights
Output	Hacker loses latest legal bid over driver rights
Seedword	sensitive company documents
Headline	Investigators revise cause of escape room fire that killed 5 girls
Output	Sensitive company documents revise cause of escape room fire that killed 5 girls
Seedword	forensical analysis
Headline	MPs’ threat to block government’s tax without second brexit referendum
Output	MPs’ threat to block forensical analysis without second brexit referendum

Figure 8: Generated headlines for the input text in Figure 4.

fulfilled sufficiently. The headlines are often grammatical, but not always. Sometimes, the dependency parser has trouble selecting the full noun phrase in both the input text and in the headlines from the headline database, which leads to only partially substituted objects and subjects. Since Churnalist is meant for supporting game writers, we rely on the user to filter and discard ungrammatical output.

In the current version of the system, where seed words and headlines are selected and matched at random, many of the generated headlines would probably not yet be considered context-appropriate. For example, readers will not necessarily relate headlines mentioning ‘company documents’ to the stolen company documents from the game text. We have not yet formally evaluated the output of our system for the ‘context-appropriate’ property. We plan on making further improvements to the system and evaluating both the system and the outputs. It could be that readers behave according to Veale’s ‘charity of interpretation’ and are more generous in

their interpretation than we anticipate.

Some seed words have a stronger connection to the game story and will evoke a stronger sense of coherence than others. For example, we expect that headlines that mention ‘hackers’ will be easier for the readers to connect to the story than headlines that mention ‘managers’. Most companies have managers; few companies have problems with malicious attacks from hackers. We expect that incorporating a stricter filter for seed words will lead to headlines with a stronger link to the game story from the input text. For example, we could rank seed words based on their term frequency-inverse document frequency (tf-idf). This would take into account that seed words that occur frequently in general are probably less representative for the input text than seed words that occur rarely in other English texts. For now, we leave the task of filtering the seed words and generated headlines to the human user of Churnalist.

Finally, choosing a random headline from the database for

substitution is a mixed blessing. On the one hand, combining a context word with the randomized headline can lead to surprising, unexpected and creative outputs. On the other hand, sometimes the link with the context word that was chosen for substitution is far-fetched or even downright ridiculous.

The application domain of Churnalist is supporting game writers in their creative task. Since Churnalist requires no linguistic knowledge, it is an accessible tool. Instead of relying on hand-written linguistic models, it requires external datasets for its text modification functionality. By using News API for collecting headlines and using the FastText dataset as knowledge base, Churnalist can run fully on publicly available data. Similarly to PoeTryMe, users can choose to use different datasets for their particular application, for example for a different language than English.

However, using external data for NLG has some caveats. Reusing headlines has as advantage that we do not have to write templates. The disadvantage is that quality of the output headlines will never be better than the quality of the headlines from the headline database. Using headlines from low-quality news outlets with click-bait headlines, the output headlines will show similar clickbait properties.

Conclusion

We have presented Churnalist, a system for generating fictional headlines. The content of the headlines is determined by the noun phrases present in the input text. Churnalist creates new headlines by taking keywords from the input as seed words. It expands the list of seed words by querying a knowledge base of word embeddings for related words and injecting these into existing headlines via word substitution. We circumvented problems with out-of-vocabulary seed words by using word vectors based on subword information. The user can fine-tune the quality of Churnalist's output by filtering the intermediary output of each step in the system pipeline.

Churnalist can be used by game writers, as an authoring aid for writing flavor text in the form of headlines. We have provided example outputs for every step in the system pipeline, generated from game text from a dilemma-based game. Since our system was developed using publicly available datasets, open source libraries and only simple text modification techniques, Churnalist requires no linguistic expertise from its users. Although Churnalist is currently implemented for English, the use of external datasets allows us to adapt the system to other languages and use cases with minimal effort. This makes Churnalist suitable for different languages and game types.

Future work

There are three main directions for future work on Churnalist. Firstly, we can improve the implementation by using better and more appropriate tools and resources. We used an open source dependency parser for Churnalist that was trained on a standard corpus of English. Training the parser on a set of headlines could improve its accuracy, which might lead to better quality text transformations. Similarly, we expect that the quality of text transformation will improve if the headline

database consists of better quality data, such as the annotated GigaWord corpus (Napoles, Gormley, and Van Durme 2012).

Secondly, there are several possibilities for expanding Churnalist's approach to generation. Like related systems (Gonçalo Oliveira 2012; Özbal, Pighin, and Strapparava 2013; Repar et al. 2018), Churnalist could use a generate-and-test strategy, where multiple candidate headlines are generated and a fitness function determines the best candidate headline or headlines from this set as output. Instead of applying word substitution to randomly chosen headlines, Churnalist could select headlines that show semantic similarity with the input text and use these as a basis for transformation. More advanced methods for keyword extraction from the input texts could be used, going beyond simple noun extraction. Using additional semantic resources, such as Wordnet or ConceptNet, could also help Churnalist in suggesting more valid words to the user.

Thirdly, we would like to expand Churnalist's outputs to also include social media messages, to make it possible to automatically generate social media messages as flavor text. To generate social media messages, we could take a similar text transformation approach as we have used for the headlines. As social media is often used to share news headlines, we can incorporate headlines as a specific type of social media messages. In fact, some malicious Twitter bots use text modification techniques and news headline sharing to disguise the fact that they are bots (Hegelich and Janetzko 2016). Additionally, both headlines and social media messages are interesting vehicles for exploring affective language generation. For example, we could generate headlines with a particular political slant or social media messages that express a particular emotion.

Finally, we still need to evaluate our approach to text generation for games. We plan to do so in various ways. We want to ask human judges to assess the output of Churnalist on properties such as grammaticality and 'context-appropriateness' (see our headline requirements), and draw comparisons with a baseline. Given the current popularity and quality of neural generation systems, we would also like to compare the output of Churnalist to state-of-the-art neural headline generation systems, given the same game text as input.

Acknowledgments

This research is supported by the Netherlands Organisation for Scientific Research (NWO) via the DATA2GAME project (project number 055.16.114). We would like to thank Dr. Lorenzo Gatti and all reviewers for their useful remarks.

References

- Backus, K. 2017. Managing output: boredom versus chaos. In Short, T. X., and Adams, T., eds., *Procedural Generation in Game Design*. AK Peters/CRC Press. chapter 2, 13–21.
- Banko, M.; Mittal, V. O.; and Witbrock, M. J. 2000. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL)*, 318–325.
- Bay, B.; Bodily, P.; and Ventura, D. 2017. Text transformation via constraints and word embedding. In *ICCC*, 49–56.

- Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2017. Enriching word vectors with subword information. *Transactions of the Association of Computational Linguistics* 5:135–146.
- Charnley, J. W.; Pease, A.; and Colton, S. 2012. On the notion of framing in computational creativity. In *ICCC*, 77–81.
- Colmenares, C. A.; Litvak, M.; Mantrach, A.; and Silvestri, F. 2015. Heads: Headline generation as sequence prediction using an abstract feature-rich space. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 133–142.
- Colossal Order. 2017. *Cities: Skylines*. Game [PC]. Paradox Interactive, Stockholm, Sweden.
- Dorr, B.; Zajic, D.; and Schwartz, R. 2003. Hedge Trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL 03 Text Summarization Workshop*, 1–8.
- Eidos Montral. 2011. *Deus Ex: Human Revolution*. Game [PC]. Square Enix, Shinjuku, Tokyo, Japan.
- Freehold Games. 2018. *Caves of Qud*. Game [PC/Mac/Linux]. Freehold Games, USA.
- Gatti, L.; Özbal, G.; Guerini, M.; Stock, O.; and Strapparava, C. 2015. Slogans are not forever: Adapting linguistic expressions to the news. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2452–2458.
- Gatti, L.; Özbal, G.; Guerini, M.; Stock, O.; and Strapparava, C. 2016. Heady-lines: A creative generator of newspaper headlines. In *Companion Publication of the 21st International Conference on Intelligent User Interfaces, IUI 2016*, 79–83.
- Gonçalo Oliveira, H. 2012. PoeTryMe: a versatile platform for poetry generation. In *Proceedings of the ECAI 2012 Workshop on Computational Creativity, Concept Invention, and General Intelligence (C3GI at ECAI 2012)*.
- Gonçalo Oliveira, H. 2017. O Poeta Artificial 2.0: Increasing meaningfulness in a poetry generation twitter bot. In *Proceedings of the Workshop on Computational Creativity in Natural Language Generation (CC-NLG 2017)*, 11–20.
- Grinblat, J., and Bucklew, C. B. 2017. Subverting historical cause & effect: generation of mythic biographies in Caves of Qud. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, 1–7.
- Hegelich, S., and Janetzko, D. 2016. Are social bots on Twitter political actors? Empirical evidence from a Ukrainian social botnet. In *Tenth International AAI Conference on Web and Social Media*.
- Hofstadter, D. 1995. Preface 4: The ineradicable Eliza effect and its dangers. In *Fluid concepts and creative analogies: computer models of the fundamental mechanisms of thought*. Basic Books, New York.
- Jing, H. 2000. Sentence reduction for automatic text summarization. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, 310–315. Seattle, Washington, USA: Association for Computational Linguistics.
- Lukin, S. M.; Ryan, J. O.; and Walker, M. A. 2014. Automating direct speech variations in stories and games. In *Tenth Artificial Intelligence and Interactive Digital Entertainment Conference*.
- Maxis. 1996. *SimCity 2000*. Game [PC]. Maxis Software Inc./Electronic Arts.
- Napoles, C.; Gormley, M.; and Van Durme, B. 2012. Annotated GigaWord. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, 95–100. Association for Computational Linguistics.
- Özbal, G.; Pighin, D.; and Strapparava, C. 2013. BRAINSUP: Brainstorming support for creative sentence generation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, 1446–1455.
- Repar, A.; Martinc, M.; Žnidaršič, M.; and Pollak, S. 2018. BISLON: BISociative SLOgaN generation based on stylistic literary devices. In *ICCC*, 248–255.
- Schlünder, B., and Klabunde, R. 2013. Greetings generation in video role playing games. In *Proceedings of the 14th European Workshop on Natural Language Generation*, 167–171.
- Shen, S.-Q.; Lin, Y.-K.; Tu, C.-C.; Zhao, Y.; Liu, Z.-Y.; Sun, M.-S.; et al. 2017. Recent advances on neural headline generation. *Journal of Computer Science and Technology* 32(4):768–784.
- Strong, C. R.; Mehta, M.; Mishra, K.; Jones, A.; and Ram, A. 2007. Emotionally driven natural language generation for personality rich characters in interactive games. In *Proceedings of the Third Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 98–100.
- Thiennot, J. O. 2013. *Cookie Clicker*. Game [PC/Browser]. <http://orteil.dashnet.org/cookieclicker/>. Played September 2018.
- Veale, T. 2016. The shape of tweets to come: Automating language play in social networks. *Multiple Perspectives on Language Play* 1:73–92.
- Yannakakis, G. N., and Togelius, J. 2011. Experience-driven procedural content generation. *IEEE Transactions on Affective Computing* 2(3):147–161.

Analyzing Art Works: The Six Steps Methodology

Luis Fernando Gutiérrez and Rafael Pérez y Pérez

Departamento de Ingeniería Industrial, Universidad de los Andes, Bogotá, Colombia
lf.gutierrez399@uniandes.edu.co

Departamento de Tecnologías de la Información, UAM Cuajimalpa, México City.
rperez@correo.cua.uam.mx

Abstract

Art analysis is a key aspect for computational systems whose goal is to generate visual artifacts. This paper proposes a six steps methodology to analyze and represent design principles from art works. Our approach starts with an image segmentation followed by the construction of a straight skeleton. Then we extract some color information and perform shape analysis and classification. Finally some design principles are calculated and groups of elements are built over a complete graph. Our internal art work representation gives us a way to approximate the phases of artistic appreciation proposed by some authors. We show a procedure to generalize compositional rules for the generation of new abstract art works based on the steps of the proposed methodology. We plan to use a self organizing map to cluster our art work representations and use this information to build a hypergraph and/or multigraph. Since these graphs can represent design principles, the system will be able to use these structures to explore new ways to generate artifacts and measure their novelty compared to previous exemplars.

Introduction

As suggested by Cetinic et al. (2018), analyzing artworks is a complex task which generally involves understanding aspects like form, content and meaning. These aspects originate from the formal elements present in the artwork such as line, shape, color, texture, mass and composition (Barnet, 2015). Art experts usually do their analysis comparing paintings to find relations between them (Seguin et al., 2016). Generally, the outcomes of those analyses can lead to style classifications, genre determinations, formal comments and influences between artists, artworks or art movements (Saleh et al. 2014, Florea et al. 2017, Badea et al., 2018).

In the last few years, research in computer vision techniques to analyze visual art has increased in quantity and quality (Badea et al., 2018). This trend depends on two facts. First, there have been consistent efforts by museums and collectionists to digitize more paintings and include relevant meta-data. This permits us to have larger datasets to do analysis. The second fact is the development of deep neural networks. Style classification has received more attention (Bar

et al., 2015; Saleh and Elgammal, 2015; Elgammal et al. 2018). Genre classification has been explored but there are still complex open challenges (Condorovici et al., 2013). The approach used in such classifiers, usually includes extracting a set of image features and using them to train different classifiers such as support vector machines, neural networks or k-nearest neighbors (Cetinic et al., 2018).

Art analysis is a key aspect for computational systems whose goal is to generate visual artifacts. Such analysis allows building knowledge structures from real pieces of art that can be exploited by creative agents to create more elaborated outputs. Thus, we need to develop mechanisms that allow computer systems to improve their “art appreciation” in general (Norton et al., 2010; Health et al., 2016). In order to accomplish this goal, we require general and specific knowledge (Barret, 2007). That is what this project is about. We are interested in studying and representing notions related to composition and design principles like balance, symmetry, size, contrast and shape (Pérez y Pérez et al. 2013; Pérez y Pérez & Guerrero 2019) from abstract pieces of visual art. *In this paper, we claim that the development of systems that allow analyzing and representing design principles from well-known pieces of art are important to generate better creative agents.* Thus, we propose a six steps methodology (6SM) that combines and advances well known algorithms for image processing in order to obtain such design principles.

This is a work in progress. Therefore, the key target of this text is to present to the reader the core aspects of our six steps methodology to represent design principles (see the section titled Art Work Representation) as well as showing some partial results. The first step consists of an image segmentation using the algorithm proposed by Syu et al. (2017) (see the Image Segmentation section). The objective of this phase is to build a hierarchical multi-resolution representation of the regions that make up an image. The second step builds a planar graph called straight skeleton over every region or segment of the previous step (see the Straight Skeleton and Centrality Measure section). The purpose of this graph is to induce a terrain model. With this model a centrality measure can be computed and a generalized notion of center of mass can be defined. The third step extracts color information (see the section titled Color Information) based

on Itten's model (1974). To achieve this phase, we follow Sartori et al. (2015) proposal to build a 180 color swatch. With this color palette we replace every original color and calculate spatial relations over Itten's color sphere generated by these 180 colors. The fourth step consists of a shape classification process (see the Shape Classification section). The objective of this step is to do a clustering procedure to have a reduced number of general shapes that can represent most of the regions of the art works analyzed. The next phase implements some binary relations between regions based on some design principles (see the Design Principles and Binary Relations section). We calculate relations on pairs of shapes based on measurements over the mean shape bounding box, direction and aspect ratio. The final step consists on building groups as suggested by (Pérez y Pérez et al. 2013; Pérez y Pérez and Guerrero 2019) (see the section titled Groups of Regions). The goal of this step is to have a representation of the elements of the art work at different levels of abstraction.

We are employing the Tlahcuilo visual composer (Pérez y Pérez et al., 2013; Pérez y Pérez & Guerrero, 2019) to implement a proof of concept. We will include new design principles and new evaluation procedures. Because having more artistic knowledge should help achieve higher quality artifacts (Heath et al. 2016), we hope to improve the quality and novelty of the artifacts the Tlahcuilo produces.

Related Work

Recently, a large number of image representations presented in the literature are exclusively or highly dependent on abstract neural network feature maps. Of particular interest to this research are works that suggest image representations such as the one proposed by Bar et al. (2014) that involve more concepts interpretable by humans directly. The authors suggest using a combination of powerful neural network visual features with other descriptors. The effectiveness of convolutional neural network based features, particularly in combination with other hand-crafted features, was confirmed also for genre classification by Cetinic and Grgic (2016).

Artistic style classification is another related problem that has been addressed with continuous increasing interest. Some recent work used object recognition (Crowley and Zisserman, 2014). The authors show that finding objects in paintings by learning object-category classifiers from available sources of natural images is possible. Artistic scene or genre understanding is also important. Badea et al. (2018) investigate the relation between genre, scene and artistic image subject. The authors investigate abstraction achieved by deep convolutional neural networks. In particular, "Abstract Art", is targeted by the authors as a challenging problem since a subject is not necessarily present.

Saleh et al. (2014) study how painters influence each other using visual similarity. They implemented a procedure based on computer vision and machine learning. The authors perform several comparisons using different visual

features and similarity measurements. Since there is not enough ground truth information to achieve influence analysis directly, the authors use a highly correlated task such as style classification to show some results. The authors investigate features aspect of the paintings and compare semantic-level features vs low-level and intermediate level features. They claim that their study confirms that high-level semantic features are more useful for style classification and hence for influence analysis. In a similar direction, Seguin et al. (2016) investigate how state-of-the-art machine vision algorithms can be used to retrieve common visual patterns shared by sets of paintings. Florea et al. (2017) suggest that visual similarity has space for improvement because most of the research and results have been developed for older artistic movements where scene depiction has high-level semantic concepts and does not present particular abstractions.

In the domain of computational creativity, DARCI (Norton et al. 2010; Heath et al. 2016) is a reference. The goal of this system is to eventually produce images through creative means. In the process to achieve this, the authors propose to teach DARCI some artistic image appreciation and understanding. They implement this through the association of low-level image features to artistic descriptions. They show that the system successfully learns 150 different descriptors from images. Pérez y Pérez and his colleagues (Pérez y Pérez et al. 2013; Pérez y Pérez & Guerrero 2019) propose a computer model to develop visual compositions based on the Engagement and Reflection Model. The system uses design principles to analyze examples provided by designers and generate a knowledge base to progress a visual work and also measure the novelty of its artifacts.

Garcia and Vogiatzis (2018) argue that to build artistic knowledge, we have to work outside the style classification tasks and expand our research goals. The authors present SemArt, a multi-modal dataset for semantic art understanding. The authors suggest a challenge called Text2Art to evaluate art understanding based on a retrieval task. They also suggest several models for encoding visual and textual artistic representations into a common semantic space. Strezoski and Worring (2018) created a large dataset with more than 430,000 samples called "The OnmiArt Challenge". They suggest analyzing more attributes related to the art works.

Art Work Representation

The knowledge structures constructed in this paper are based on examples of abstract art created by human artists. Since abstract art can be thought of as lacking representation of common everyday objects, these art works are more prone to intrinsic artistic formal aspects. Initially we propose to develop an analysis using exclusively 165 of Rothko's art works. We have a partial prototype that implements the pipeline described in the next few sections. It receives an image as input and outputs our internal representation.

Step 1: Image Segmentation

Syu et al. (2017), claim that even though plentiful segmentation algorithms have already been proposed, how to effectively partition an image into segments that are “meaningful” to human visual perception is still very challenging. Sometimes it is not enough to choose a specific image characteristic such as color or texture to achieve a successful image segmentation. This paper proposes to use the new algorithm developed by Syu et al. (2017) which builds a hierarchical image segmentation. One of the objectives of the algorithm is to generate a dendrogram in which every node corresponds to a segment and all the nodes in the same level build up a segmentation. This dendrogram is a consistent multi-resolution representation of the contents of the segmented image. In our context, consistent means that every new segment comes exclusively from a previous node. This characteristic is one of the main differences between the algorithm developed by the authors and similar hierarchical solutions.

The algorithm proposed by Syu et al. (2017) works in two steps like almost all hierarchical procedures. The first phase works directly with the raw pixels and has the objective to group up similar pixels into regions. For the second step of the first phase of the algorithm, the authors present an iterative process of contraction and merging. The contraction step is based on an optimization process over an affinity matrix that groups pixels. The merging is characterized by a fixed grid in which previously contracted pixels are joined together. These two steps work exclusively on the similarity at color level between adjacent pixels.

The second phase keeps on making a contraction and merging procedure. The affinity matrix gets updated based on color, size, texture and intertwining of the regions. The affinity matrix for the second phase depends on the dissimilarity between regions R_i and R_j . The factors that make up the dissimilarity metric are the following:

Color Component: For a region R_i , Syu et al. (2017) denote its color feature as the averaged color values inside the region. For adjacent regions R_i and R_j , the color-based dissimilarity measure is $D_c(R_i, R_j) = \|C_{R_i} - C_{R_j}\|_2$.

Texture Component: two adjacent regions with similar texture patterns and similar colors should have a larger affinity value. To describe the texture pattern of a region, Syu et al. (2017) convert the region color values to gray and calculate the Weber Local Descriptor (WLD). This descriptor consists of two components, differential excitation and orientation, over a local window around every pixel.

Region Size Component: To take into account the region size in the merging process of the second phase, the authors define an additional distance function is design to facilitate the merging of two small regions or the merging of small regions into their neighboring regions fast.

Spatial Intertwining Component: this component is used to merge together small regions produce during the cycles. Syu et al. (2017) measure the intertwining of any pair of regions i and j based on a fixed 5x5 window over every pixel p of R_i . They calculate the most common index in the local

window. Based on these indexes, a decision to merge smaller groups of pixels to neighboring regions is taken. Figure 1 and 2 give examples of the result of the first phase of the process we are describing. We show the hierarchical segmentation procedure achieved by our partial prototype applied to one of Rothko’s and Kandinsky’s artworks.

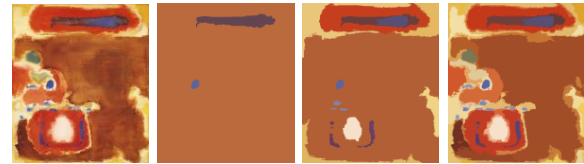


Figure 1 – original, 3, 13 and 60 segments



Figure 2 – original, 2, 13 and 60 segments

Step 2: Straight Skeleton and Centrality Measure

According to Huber (2012), the notion of a straight skeleton for a simple polygon P was introduced for the first time by Aichholzer et al. (1995). The authors used a wavefront propagation process to define it. As reported by Huber (2012), every edge e of P sends out a wavefront which moves inwards at unit speed and parallel to e . Figure 3 shows a visualization of the described process. During the wavefront propagation process, topological changes named events are produced. Huber distinguishes two types: edge and split events. An edge event occurs when two neighboring convex vertices u and v of the wavefront meet. This event causes the wavefront edge e , which connects u and v , to collapse to zero length. The wavefront edge e is removed and the vertices u and v are merged into a new convex vertex. A split event occurs when a reflex vertex u of the wavefront meets and edge e of the wavefront. The vertex u splits the entire wavefront into polygonal parts (Huber, 2012).

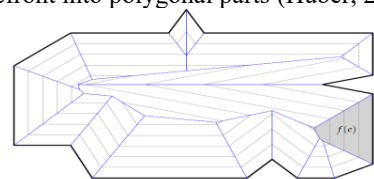


Figure 3 – visualization of the wavefront propagation process. The edge e has the associated front wave $f(e)$. Image reproduced from Huber (2012), pg. 13.

The formal definition of a straight skeleton taken from Huber (2012) is as follows: the straight skeleton $S(P)$ of a polygon P corresponds to the straight segments that are traced out by the vertexes of any wavefront. These segments are denominated arcs or edges of $S(P)$. The places where topological changes take place are defined as nodes. To every edge e of P belongs a face $f(e)$, which consists of every point traced by the wavefront border started by edge

e. Every node of $S(P)$ is incident to multiple arcs. The border of any given face consists of the arcs and vertices of $S(P)$.

The straight skeleton has several interesting properties. Central to this work is the fact that this skeleton is a tree (Aichholzer et al., 1995). Based on that fact, we know there exists a unique node that can be named as the root. The same authors generalized the concept of straight skeletons to planar straight-line graphs. According to Huber (2012), this generalization allowed a better interpretation of the visualization suggested initially by Aichholzer et al (1995). This intuition is formalized with the definition of model terrain: the terrain $T(G)$ of G a straight-line graph is: $T(G) = \bigcup_{t \geq 0} W(G, t) * \{t\}$ where $W(G, t)$ corresponds to the wavefront of a straight-line graph G for a time $t \geq 0$. According to Huber (2012), $W(G, 0)$ corresponds geometrically to the same graph G and it should be visualized like a superposition with the same topology over the original graph.

The straight skeleton induces an Euclidean graph given by the spatial coordinates of the nodes. Taking into account the terrain model induced by the straight skeleton and the distance between every node, a weight for every edge is assigned. The value is just the multiplication of the $t \geq 0$ parameter or third dimensional value of $T(G)$ with the distance between every pair of nodes. We calculate a centrality measure based on closeness and using the weights as cost function to find a unique node that can represent each region. Since the straight skeleton is a tree, we can be certain that such a unique node always exists. The center of this Euclidean graph, can represent each region and acts as a generalized notion of a mass center. Figure 4 shows the terrain model induced by the straight skeleton of a region and visualizes the center of the Euclidean graph.

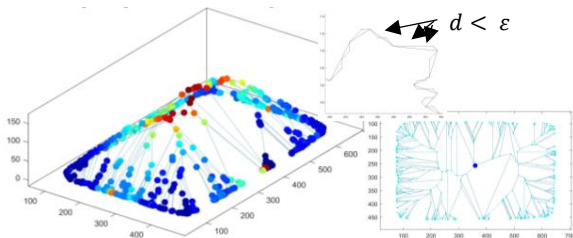


Figure 4. Terrain model induced by a straight skeleton. Center of Euclidean graph based on closeness centrality measure.

The process by which the straight skeleton is generated is computationally demanding. To cope with this time and the large number of regions used in this paper, we propose to make a region simplification based on the Ramer–Douglas–Peucker algorithm. The objective of the algorithm is, given a linear segment curve, to find an approximated curve with less points. The dissimilarity measure used to compare both curves is based on the Hausdorff distance. One of the most important characteristics of the procedure is that the approximated curve points, come from a subset of the original curve points. For the algorithm to work, a tolerance parameter $\epsilon > 0$ needs to be given by the user. After some trial

and error, we found $\epsilon = 0.1$ to be a good candidate. The first step of the algorithm is to find the farthest points on the original curve. Once both points of the maximum diameter are identified, the region is split in two curves and the algorithm is invoked recursively on both segments. The original two points are automatically assigned as elements of the new approximated curve. The next step is to find the farthest point between the straight line made up by the two original points and every other point of the segment. Once this point is identified, the distance between the line and the point is calculated. If the distance is less than ϵ then any given points between the original points and the farthest point get eliminated. This guarantees that the tolerance requirement is met in every step. In case the distance is greater than ϵ , the algorithm marks this new point as belonging to the new approximated curve. The algorithm gets recursively invoked between the new subsegments. The recursion process is over when there are no more segments to check. Setting $\epsilon = 0.1$ allows us to simplify every border region without losing important perceptual geometrical characteristics. After the simplification based on the Ramer–Douglas–Peucker algorithm, we order all the components of every segmentation in every level. In case we find more than one component, we only calculate the straight skeleton over the largest component. If holes are found inside any of the regions, we calculate their size and include them only if they represent more than 1% of the original containing region.

Step 3: Color Information

Most of the color information is going to be based on Itten’s theory (1974). Itten studied and taught almost all aspects related to aesthetic and expressivity of color during several years in the Bauhaus school. His theory defined a set of rules for colors and combinations of them. These principles were named by the author as “objective principles of color”. Additionally, Itten also tried to formalize some of the emotional aspects of color combinations. The author’s theory is visually represented by a color wheel. This structure is composed of 12 color shades made out of primary, secondary and tertiary colors. Colors that are opposed in the color wheel are complementary and make up a harmonic pair (Sartori et al., 2015). Itten’s wheel was further expanded using 5 levels of luminance and 3 levels of saturation. This last model, composed of 180 colors was named by the author as Itten’s color sphere (Sartori et al., 2015).

In order to condense and reduce color information, we implement a color swatch construction process, based on what Sartori et al. (2015) suggested. The idea of this process is to sample all the colors out of a dataset and use a cluster process like k-means to find 180 centroids in RGB color space. In our case, the dataset is based on all the color regions of every hierarchical segmentation level. The similarity metric used by the k-means algorithm in our case is the ordinary three dimensional Euclidean distance. Once the 180 centroids are found (Figure 5), we proceed to replace every

color with the nearest centroid to change every color in terms of the newly built color swatch (Figure 6).

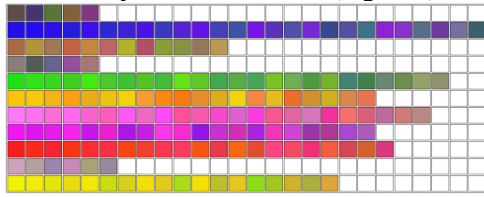


Figure 5. Color palette – 180 colors. 11 groups of colors. Up to down: black, blue, brown, grey, green, orange, pink, purple, red, white and yellow.



Figure 6. Color palette application.

The color relationships used in this paper are going to be binary and related to contrast and harmony. For the first relation, we simply use Itten’s color sphere and try to locate opposite elements. For harmony relations, we suggest using the Itten’s relations that correspond to defined geometric patterns over the sphere or color wheel. The color swatch also permits us to establish some groups of colors that give global information of the subset of colors from the swatch that are present in an artwork.

Step 4: Shape Classification

The definition of the concept of “shape” has always been complicated. According to Dryden & Mardia (2016), in the ordinary and common use of the word “shape”, we almost always use it in an indirect way and using relations of similarity to other objects to try to specify a specific “shape”. Dryden & Mardia (2016) present the following definition: “shape is all the geometrical information that remains when location, scale and rotational effects are removed from an object” (pg. 22).

Based on this definition, the shape of an object is invariant under any Euclidean similarity transformation. Two objects have the same shape if one of them can be translated, rescaled and rotated in such a way that superimposing it over the other one, they match completely. The way in which a shape is represented is fundamental to developing any analysis. This work is going to use a finite set of points over the straight skeleton to build some pseudo-landmarks. The configuration matrix X is the $k \times m$ matrix of Cartesian coordinates of the k landmarks in m dimensions. In particular, we are going to work with $k \geq 3$ landmarks and the dimension m will be 2. Kendall (1984) demonstrated that for the particular case in which m is 2, the shape space is a Riemannian manifold (complex projective space). In particular and for the purposes of this paper, we need to use the Riemannian distance to be able to compare the difference between any

two shapes once any translation, scale or rotation is removed.

To classify the regions generated by the hierarchical segmentation algorithm, we suggest a process that builds several configuration matrixes. The construction starts with $k = 12$ and ends as soon as one of the following two conditions are met: a) the maximum number of points of the region is less than 50, b) $alr/alo \geq 0.90$ and $1 \geq ar/ao \geq 0.90$, where alr is the arc-length of the approximated curve, alo is the arc-length of the original curve, ar is the area of the approximated curve and ao is the area of the original curve.

We compare every pair of regions based on a 12-points configuration matrix and gradually increase the number of configurations if necessary. We use the Ramer–Douglas–Peucker algorithm to simplify every region on our dataset to find the respective configuration matrixes. In applying the algorithm, we use the notion of tortuosity: $\tau = \frac{L}{C}$ where L corresponds to the length of the curve and C corresponds to the distance between the extreme points. Since every region is closed, we start by splitting every border by the major axis and using the tortuosity measure to decide what sub curve has more complexity and start the procedure. On every step, we compare the tortuosity of every subsegment to decide the next candidate. Figure 7 shows some steps of the above procedure for one region. In case a region has less than 12 points, we artificially interpolate points between the longest segments until necessary.

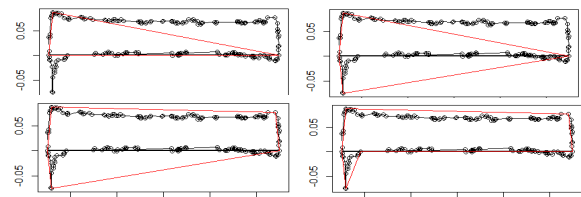


Figure 7. Region 2 of Rothko’s “Violet, Black, Orange, Yellow on White and Red” (1949) work. 3, 4, 5 and 6 points simplification in red.

To classify these simplified regions, we do a clustering procedure over all the 12 points configuration matrixes using the ideas of Vinué et al. (2014). The authors suggest an extension to the original k-means algorithm so that it can be applied directly over configuration matrixes using the Procrustes analysis and the Riemannian distance. Starting with 1553 simple regions, we suggest to find initially 90 centroids on this first step. In the second step of our procedure, we do a hierarchical clustering with single linkage to have the possibility of reducing further the number of clusters. To try to identify where the hierarchical tree should be cut, we analyze the distance matrix calculated with the Riemannian distance function between all the centroids. After some experimentation using between 2 and 10 nearest neighbors of every element in this matrix, we found that taking 25% of the maximum distance between any element was a good

condition to stop the pruning procedure. We check the condition “b” stated before ($\frac{alr}{alo} \geq 0.90$ and $1 \geq \frac{ar}{ao} \geq 0.90$) to decide if every centroid is a good representative of the cluster. In case there is a configuration region which does not fulfill this condition, we propose to split the respective cluster augmenting the configuration matrixes one step at a time until condition “b” is true. After doing this procedure, the hierarchical clustering tree allows us to move up or down in the representation of any region. If we want to simplify the configuration matrix to generate simpler regions, we use the tree and all the members of a cluster to pick a simpler configuration matrix or generate a new mean shape. The final centroids act as the color swatch we defined before and is going to allow us to generalize some binary relations mentioned in the next section. Figure 8 shows a contrast relation between two regions and gives us an idea of how a generalization of this relation is possible.

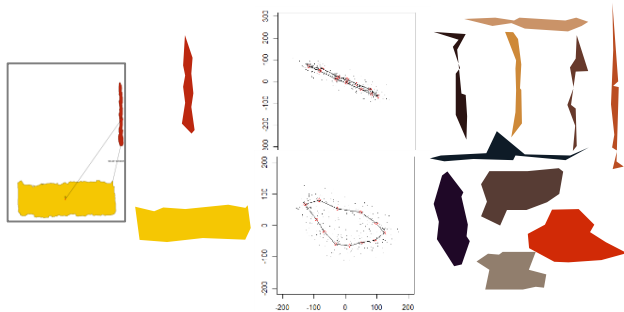


Figure 8. Contrast binary relation – directional, size and proportion. 12-points simplification, cluster centroid and similar regions for relation generalization.

Step 5: Design Principles and Binary Relations

Based on the output of the previous step, we are going to define some contrast binary relations between the centroids. After a linear transformation that takes the end points of the major axis of every centroid to the coordinates $(-\frac{1}{2}, 0)$ and $(\frac{1}{2}, 0)$ we calculate the mean shape bounding box, aspect ratios, distance and mean direction. To define this direction, we propose to use a histogram based on the shortest path that connects the major axis end points and passes through the center of the straight skeleton (Figure 9). We suggest having 6 bins in the histogram. Every bin is 30° and the range starts in -15° . Once the histogram is normalized, we can have an approximation to the orientation of the longest path and use this as a general notion of direction for the region.

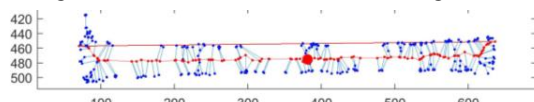


Figure 9. Major axis and shortest path on SS.

This work also proposes to calculate balance, symmetry and rhythm relations as suggested by Pérez y Pérez et al.

(2013). We also propose to expand the use of contrast from color (Pérez y Pérez et al. 2010) to shape and incorporate more design principles than the original work developed by the authors. We believe the region representation suggested will allow us to generate even more art principles in future works since the straight skeleton lets us calculate internal symmetries of the shape, proportions and more. In Figure 10 we show some examples of binary relations found using our partial prototype.

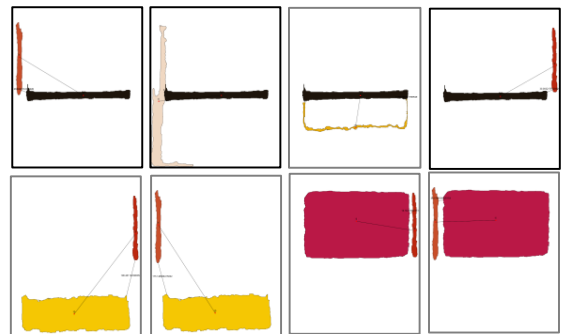


Figure 10. Binary relations – directional, size and proportion contrast examples. Balance and Symmetry.

Step 6: Groups of Regions

Based on the ideas presented in (Pérez y Pérez et al. 2013; Pérez y Pérez and Guerrero 2019), we build groups of regions based on the distance. We construct a complete graph of the first 20 regions of an art work using the normalized center of every region (Figure 11). The order in which the graph is built is based on the size. We calculate the distance between every pair of nodes. The final weight of every edge is the normalized Euclidean distance by the main diagonal of the image of the art work. To start building the groups of layer 1, we follow the procedure described by Pérez y Pérez et al. (2013) and iterate their procedure until no more groups or layers are possible.



Figure 11. Rothko’s “Violet, Black, Orange, Yellow on White and Red” (1949). 10 simple regions and complete graph (SS).

Internal Representation

The goal of the internal representation is to extract the information of the art work related to the design principles and binary relations suggested for the analysis. We suggest a categorical-numerical vector that represents the art work with the following structure:

- Global information: artist name, year of creation and the artistic style of the art work.
- Specific art work information: width, height, aspect ratio, first 20 simple regions based on the dendrogram of

- segmentation ordered by area in decreasing order. In case there are less regions, values are filled with zeros.
- c) Region specific information: every region is represented by the following attributes: area of the region normalized by the total area of the art work, centroid identifier of the cluster to which this region is closer based on Riemannian distance, normalized width of the bounding box of the region, histogram of direction and color identifier from the color swatch.
 - d) Groups of Regions: information based on the groups constructed using the procedure described previously.
 - e) Relations between Regions: using the complete graph induced by the center of every one of the simple regions, we propose to analyze every possible binary relation between every pair of regions. Existence of the respective binary relation between any pair of simple regions is represented by 1 or 0. The order of the attributes is based on literal b. Every binary relation is weighted by the distance between the nodes of the complete graph.

Discussion and Future Work

In this paper we have introduced what we refer to as the Six-Steps Methodology (6SM) to analyze and represent design principles from well-known pieces of abstract visual art. The knowledge representation was built taking into account some important aspects of the appreciation and perception processes such as color, orientation, shape, proportion, contrast size and grouping (Liu et al., 2017). In the near future, we plan to include texture more explicitly and work some design principles based on it. The hierarchical segmentation gives us a way to approximate the phases of artistic appreciation proposed by Tinio (2013) and Leder (2013). In particular, the first simple regions can capture the initialization phase mentioned by Tinio, while the rest of the levels of the segmentation can be associated to the expansion, adaptation and finalization phases proposed by the author. And since our representation allows us to get a global structure, recognize and simplify shapes, build groups and get further details as needed, we think that some of Leder's ideas (2013) related to knowledge, familiarity, content and style processing stages are captured too. We believe these facts are important for the development of computational creative systems that generate visual artifacts, because they allow them to analyze information better and constantly move between general and specific knowledge, that is necessary throughout the evaluation of the creative process. We hope to be moving in the right direction to improve the artistic appreciation of our system so it can be even more autonomous.

We have shown a procedure to classify shapes that allow us to generalize compositional rules for the generation of new abstract art works that impact directly the novelty of the artifacts generated by the system. As far as we know there are no other systems capable of obtaining, from human made abstract visual art, design principles that then can be

used for generating new pieces. Based on the work reported in this paper, we expect to be able to incorporate all these algorithms into the agent Tlahcuilo visual-composer in the following months. Then, we will be able to test the quality of the new products generated by the system and hopefully produce more results to support our view.

Besides describing the core components of the 6SM, we also suggest that the work reported here can be useful as a base to produce more robust systems. For instance, we plan to use a self organizing map to cluster our art work representations, probably using a variation of the Growing Hierarchical Self Organizing Map (GHSOM) (Raubert et al., 2002). With the cluster's information and temporal data from the art works, we suggest to build a hypergraph and/or a multigraph. Because these graphs can represent design principles, the system will be able to exploit such information, find correlations and explore new ways to generate new pieces (Hackett 2016).

The information represented by these graphs might be also useful to improve the automatic evaluation of visual pieces. For instance, it is possible to compare the hypergraph of a creative agent's products against some recognized visual art. Based on the GHSOM we can check how many previous examples share similar compositional principles and guide the creative agent's composition process depending on these results. In a similar way, depending on the distance to other nodes in the hypergraph and/or the multigraph, we could evaluate novelty. Finally, we believe these structures can be used by the system to give some explanations as to why a partial or finished composition is interesting, what design principles it is based on, what previous established rules it might break and probably to what style or styles it belongs.

We are also comparing the results of some state-of-the-art style classification neural networks feature maps with our art work representation to see if it is possible to improve the performance of those models or ours. We require more experiments to produce more hypotheses about how computational systems can generate interesting abstract artifacts that are grounded in an artistic context. The methodology we describe here is one step towards answering deeper questions about how a system uses previously available knowledge in the quest for producing new creative visual artifacts.

References

- Aichholzer, O., Alberts, D., Aurenhammer, F. and Gärtner, B. 1995. Straight Skeletons of Simple Polygons. In Proceedings of the 4th International Symposium of LESMARS, 114-124. 1995
- Badea, M., Florea, C., Florea, L., & Vertan, C. 2018. Can we teach computers to understand art? Domain adaptation for enhancing deep networks capacity to de-abstract art. *Image and Vision Computing*, 77, 21-32.
- Bar, Y., Levy, N., & Wolf, L. 2014. Classification of Artistic Styles Using Binarized Features Derived from a Deep Neural Network. *Computer Vision - ECCV 2014 Workshops Lecture Notes in Computer Science*, 71-84.

- Barnet, S. 2015. A short guide to writing about art. Pearson.
- Barrett, T. 2007. Teaching Toward Appreciation in the Visual Arts. *International Handbook of Research in Arts Education* Springer International Handbook of Research in Arts Education, 639-656.
- Cetinic, E., Lipic, T., & Grgic, S. 2018. Fine-tuning Convolutional Neural Networks for fine art classification. *Expert Systems with Applications*, 114, 107-118.
- Condorovici, R. G., Florea, C., & Vertan, C. 2013. Painting Scene Recognition Using Homogenous Shapes. *Advanced Concepts for Intelligent Vision Systems Lecture Notes in Computer Science*, 262-273.
- Crowley, E. J., & Zisserman, A. 2014. In search of art. In *ECCV workshops: 1* (pp. 54–70). Springer.
- Dryden, I. L., and Mardia, K. V. 2016. *Statistical shape analysis with applications in R*. Chichester: John Wiley & Sons.
- Elgammal, A., Bingchen, L., Mohammad, E., & Marian, M. 2017. CAN: Creative Adversarial Networks Generating “Art” by Learning About Styles and Deviating from Style Norms. Paper presented at the 8th International Conference on Computational Creativity (ICCC), Atlanta, GA, USA, June 19–23.
- Elgammal, A., Mazzone, M., Liu, B., Kim, D., & Elhoseiny, M. 2018. The Shape of Art History in the Eyes of the Machine. arXiv preprint arXiv:1801.07729 (2018).
- Florea, C., Badea, M., Florea, L., and Vertan, C. 2017. Domain Transfer for Delving into Deep Networks Capacity to De-Abstract Art. *Image Analysis Lecture Notes in Computer Science*, 337-349.
- Garcia, N. and Vogiatzis, G. 2018. How to Read Paintings: Semantic Art Understanding with Multi-Modal Retrieval. arXiv preprint arXiv:1810.09617 (2018).
- Hackett, P. M. 2016. *Psychology and philosophy of abstract art: Neuro-aesthetics, perception and comprehension*. London: Palgrave Macmillan.
- Heath, D. and Ventura, D. 2016. Before a computer can draw, it must first learn to see. In *Proceedings of the 7th International Conference on Computational Creativity*, 2016.
- Huber, S. 2012. *Computing straight skeletons and motorcycle graphs: Theory and practice*. Aachen: Shaker.
- Itten, J. 1974. *The art of color: The subjective experience and objective rationale of color*. Wiley.
- Kendall, D. G. 1984. Shape manifolds, Procrustean metrics and complex projective spaces. *Bulletin of the London Mathematical Society*, 16, 81–121.
- Leder, H. (2013). Next steps in neuroaesthetics: Which processes and processing stages to study? *Psychology of Aesthetics, Creativity, and the Arts*, 7(1), 27-37.
- Liu, J., Lughofer, E., & Zeng, X. (2017). Toward Model Building for Visual Aesthetic Perception. *Computational Intelligence and Neuroscience*, 2017, 1-13
- Norton, D., Heath, D. and Ventura, D. 2010. Establishing Appreciation in a Creative System. In *Proceedings of the 1st International Conference on Computational Creativity*, 2010.
- Pérez y Pérez, R., Aguilar, A., & Negrete, S. (2010). The ERI-Designer: A Computer Model for the Arrangement of Furniture. *Minds and Machines*, 20(4), 533-564.
- Pérez y Pérez, R., María González de Cossío, M. and Iván Guerrero, I. 2013. A Computer Model for the Generation of Visual Compositions. *Paper presented at the 4th International Conference on Computational Creativity (ICCC)*, Sydney, Australia.
- Pérez y Pérez, R. and Guerrero I. R. (2019). A computer agent that develops visual compositions based on the ER-model. *Annals of Mathematics and Artificial Intelligence*.
- Rauber, A., Merkl, D., and Dittenbach, M. 2002. The growing hierarchical self-organizing map: Exploratory analysis of high-dimensional data. *IEEE Transactions on Neural Networks*, 13(6), 1331-1341.
- Saleh, B., Abe, K., Arora, R. S., & Elgammal, A. 2014. Toward automated discovery of artistic influence. *Multimedia Tools and Applications*, 75(7), 3565-3591.
- Saleh, B., Elgammal, A. 2015. Large-scale classification of fine art paintings: Learning the right metric on the right feature. arXiv preprint arXiv:1505.00855 (2015).
- Sartori, A., Culibrk, D., Yan, Y., and Sebe, N. 2015. Who’s Afraid of Itten. *Proceedings of the 23rd ACM International Conference on Multimedia - MM 15*, 311-320.
- Seguin, B., Striolo, C., Dilenardo, I., & Kaplan, F. 2016. Visual Link Retrieval in a Database of Paintings. *Lecture Notes in Computer Science Computer Vision – ECCV 2016 Workshops*, 753-767.
- Strezoski, G., and Worring, M. 2018. OmniArt. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 14(4), 1-21.
- Syu, J., Wang, S., and Wang, L. 2017. Hierarchical Image Segmentation Based on Iterative Contraction and Merging. *IEEE Transactions on Image Processing*, 26(5).
- Tinio, P. P. L. (2013). From artistic creation to aesthetic reception: The mirror model of art. *Psychology of Aesthetics, Creativity, and the Arts*, 7(3), 265-275.
- Vinué, G., Simó, A., and Alemany, S. 2014. The k-means algorithm for 3D shapes with an application to apparel design. *Advances in Data Analysis and Classification*, 10(1), 103-132.

“She Offered No Argument”: Constrained Probabilistic Modeling for Mnemonic Device Generation

Paul M. Bodily, Porter Glines, and Brandon Biggs

Department of Computer Science

Idaho State University

921 S. 8th Ave, Pocatello, ID 83209 USA

bodipaul@isu.edu, glinport@isu.edu, biggbran@isu.edu

Abstract

A common aspect to creativity as described by creative theorists is the juxtaposition and balance of two opposing qualities, namely novelty and typicality. Practical models of computational creativity are needed that effectively leverage the contributions of each of these qualities in a synchronous manner. We discuss the effectiveness of constrained probabilistic models in representing this duality in generative models of creativity. We illustrate constrained Markov models as an example of a constrained probabilistic model and demonstrate its application to computational creativity in the elaboration of a system called NhMMonic for generating mnemonic devices. We demonstrate the effectiveness of the system¹ using a qualitative survey. Our findings suggest that the constrained Markov model is particularly effective at generating mnemonics that exhibit novelty and typicality in grammatical and semantic flow with the overall result of more effective mnemonics for the purpose of memorization. Source code as well as our mnemonic device generator are both freely accessible online.

Introduction

Computational creativity (CC) has been defined as “the philosophy, science and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviours that unbiased observers would deem to be creative” (Colton and Wiggins 2012). The plural focus on the philosophy, science and engineering of computational systems has yielded valuable theoretical contributions as well as a number of functional creative systems. Emergent from this plural focus is the challenge of maintaining harmony between theory and practice. To be sure the abstract philosophy and concrete engineering can and should work to challenge one another in their mutual growth and evolution; however, the goal ultimately is to develop systems that accurately reflect the philosophical moorings and to advance theories whose tenets agree with what is observed about creativity in practice. Thus the role of *practical models* of creativity becomes significant—models that, by virtue of their ability to implement principles deriving from the philosophy, can be gener-

¹An interactive demo can be viewed at <https://ccil.cs.isu.edu/projects/mnemonic/>

alized beyond any single creative system with great effect, while maintaining ready applicability and implementability. As described by Jordanous (2016), these models define the *creative process* of a system, namely “what the creative individual does to be creative.”

Several examples of practical models of creativity have been demonstrated. Evolutionary models represent a practical implementation of the widely-accepted theory that creativity is a self-evaluative, iterative process as discussed by Csíkszentmihályi (1996) (e.g., see Morris et al. (2012)). Related is the model of a dynamic knowledge base (Pérez y Pérez and Sharples 2004) in which novel artefacts that have been evaluated as belonging to the domain are added to a system’s set of exemplars, possibly altering the definition of the domain itself (e.g., as discussed by Boden (2003)). Generate-and-check is another model that has been suggested as being representative of the creative process (Pease, Guhe, and Smaill 2010).

In considering the modeling of theoretical aspects of creativity, one particularly intriguing aspect that is often discussed is the tenuous balance that a creative system must maintain between novelty and typicality—the adherence to structural domain-defining rules combined with an exploratory discovery of new, valuable artefacts. These two characteristics can sometimes seem at odds with one another; a creative system must both obey norms at some level and break them entirely at other levels. It is the juxtaposition of these qualities that evokes the perception of creativity: the observer recognizes and appreciates an artefact relative to its contextual domain while at the same time being challenged and surprised as a result of the artefact’s unique traits and value. Csíkszentmihályi (1996) emphasizes that creativity stems from a person learning the rules of and basic procedures of a domain and then channeling thinking based on those rules in new directions. Saunders and Gero (2001) puts novelty and typicality on a spectrum called the Wundt curve or “hedonic function” and frames successful creativity in terms of finding the correct balance of typicality and novelty (see Figure 1). Margaret Boden (2003), in her seminal work *The Creative Mind: Myths and Mechanisms*, compares (exploratory) creativity to navigating a “structured conceptual space” to find “things you’d never noticed before.” Wiggins (2006) elaborates a formal mechanism of Boden’s concept of creativity by defining two rule sets, \mathcal{R} and \mathcal{T} . Of

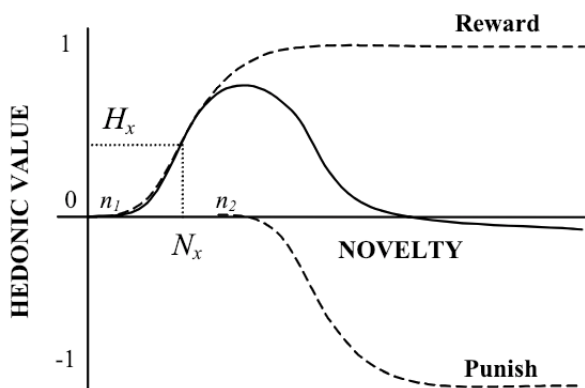


Figure 1: The Wundt curve models value as the sum of two nonlinear functions: H_x which rewards novelty, and N_x which punishes novelty beyond some threshold of typicality, from Saunders and Gero (2001).

these two sets \mathcal{R} is a set of rules which “constrain the space” to a representation of “the agreed nature of what the artefact is, in the abstract”; \mathcal{T} , by contrast, is a set of traversal rules which, when constructed effectively, is designed to find concepts that have not been previously discovered. Ritchie (2007), in defining empirical criteria for attributing creativity to a computer program, defines three essential properties, two of which are novelty and typicality (the third is quality, which Boden also emphasizes and which we will discuss below).

Many existing abstract frameworks for building creative systems have been described, several of which explicitly model the components of novelty and typicality (e.g., (Ventura 2017)). Our purpose is not to present a new framework or pattern for creative systems; rather our purpose is to discuss from an *implementation* standpoint how typicality and novelty can be modeled so as to explicitly leverage their unique contributions and simultaneously ensure that both are effectively achieved. In what follows we examine the suitability of a previously unexplored model in CC—a *constrained probabilistic model*—for this purpose. We describe how the dual nature of this model mirrors the dual properties of typicality and novelty and how the model strikes an appropriate balance between them. As a concrete example of the effective application of these models to generate novelty and typicality, we describe an implementation of a constrained Markov model, NhMMonic, for generating mnemonic devices. We show using evaluative surveys that the system generates mnemonics that demonstrate typicality, novelty, and value (as measured by how well the mnemonic facilitates memorization and learning).

Parallels Between Computational Creativity and Constrained Probabilistic Modeling

Computational creativity can be thought of as a generative act in which, for some particular domain, the set of possible artefacts $\mathcal{D} = \{x_1, \dots, x_n\}$ is represented as a random

variable X that with probability $P(x_i)$ takes on the value x_i . The primary strength of probabilistic models is that they generalize well from a set of training examples to be able to generate novel artefacts. Inasmuch as this generalization is accomplished independent of the biases of the system designer, it lends strength to the argument that probabilistic systems possess some degree of autonomy beyond manually-crafted rule-based systems. In practice, implementing a creative system in this manner presents two challenges.

One challenge is determining the probability distribution $P(X)$: with what probability should the model generate a particular x_i ? This challenge can be solved explicitly—as in the case of systems that manually encode a generative process—or implicitly—as in the case of systems that attempt to learn abstract statistical properties from a set of training examples.

Prior to or in the course of resolving the first challenge, we face a second, more formidable challenge: defining the domain \mathcal{D} itself. Decisions about whether a particular artefact x_j belongs or does not belong to \mathcal{D} can vary from one individual to the next (Koren 2010). For now let us assume that \mathcal{D} exists as a “fuzzy” subset of some larger domain, which we shall call $U_{\mathcal{D}}$ and which represents the universal set of all artefacts that can be represented using the same language with which artefacts in \mathcal{D} are represented. For example, the domain of haiku exists as a subdomain of natural language generally. The domain of musical chorales exists as a subdomain of musical compositions generally. The fuzziness of the set \mathcal{D} can derive from a variety of issues such as the difficulty in precisely defining \mathcal{D} or the willingness of domain experts to accept artefacts that (to varying extents) break the rules typical of an artefact in \mathcal{D} .

Any particular creative system defines a set that more or less approximates \mathcal{D} and possibly includes some artefacts that are less commonly agreed upon as belonging to \mathcal{D} (see Figure 2). How this set is implemented is important in designing creative systems that efficiently generate artefacts in \mathcal{D} . For rule-based systems, the rules by which an artefact belongs within the set are hard-coded; logic is designed to prevent consideration of artefacts that break rules of the domain beyond some threshold. For evolutionary models, this set can be defined by designing a fitness function that penalizes artefacts outside of this domain. The set can also be defined as a set of constraints given as input to a constraint satisfaction solver, but with limited sense of how good one solution is with respect to another (Onarheim and Biskjaer 2017).

In the process of generalization, probabilistic models trained with artefacts from \mathcal{D} are typically capable of generating artefacts that do not belong in \mathcal{D} . Increased expressive power in these models (i.e., the ability to generalize novel solutions) derives from maximizing independence relationships between elements of an artefact (e.g., being able to model rhythm and pitch separately in a music composition). This process can, however, lead to the generation of artefacts whose combined elements produce artefacts that most would agree do not belong in \mathcal{D} .

Suboptimal solutions exist to ensure that a probabilistic

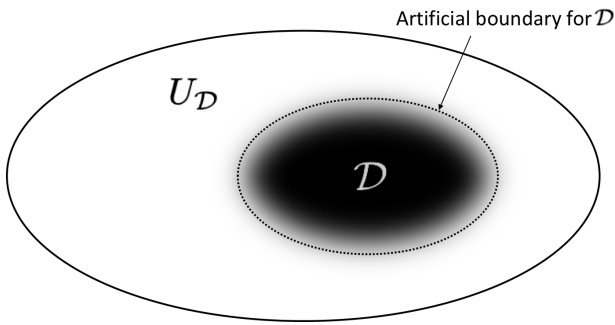


Figure 2: In many forms of creativity, the set of domain artefacts \mathcal{D} exists as a structured subset of a larger domain $U_{\mathcal{D}}$ of all artefacts that can be represented using the same language as is used to describe artefacts in \mathcal{D} . Due to the inherent difficulty of defining belonging to a particular domain for a general audience, the set of artefacts included in \mathcal{D} is in reality somewhat vague. In practice creative systems define a set that approximates \mathcal{D} which defines the expressive range of the model. The extent to which this set includes or excludes artefacts that are commonly accepted as belonging to \mathcal{D} controls how conservative or liberal the model will be in judging whether or not an artefact is representative of the domain.

model generates artefacts within the domain D of interest. Probabilistic models *could* ensure their output by minimizing independence assumptions (i.e., forcing the model to generate solutions more similar to the training data). This solution significantly decreases the model’s ability to discover novelty from the training data. This solution also requires training on data that is more precisely representative of \mathcal{D} . A second suboptimal solution is the generate-and-check or rejection sampling model: probabilistically generate artefacts using the over-generalized model and then filter results to those within the D (Pease, Guhe, and Smail 2010). This solution not only creates inefficiencies, but often assigns low probability to artefacts belonging to \mathcal{D} (Ventura 2017). In such cases it becomes improbable that the system generates valid artefacts in reasonable time (Pachet, Roy, and Barbieri 2011).

A better solution to the problem of enforcing the model’s domain of artefacts is the incorporation of constraints into a model that maintains probabilistic reasoning. The “fundamental entwining of constraints and creativity” has been noted as an area of recent interest for creativity research, “with skillful and innovative handling of constraints seen as a prerequisite for apt creative performance” (Onarheim and Biskjaer 2017).

A constrained probabilistic model defines a set of rules for belonging in \mathcal{D} as a set of constraints \mathcal{C} . Given \mathcal{C} and a probability distribution $P_{U_{\mathcal{D}}}(x_j)$ for all artefacts in $x_j \in U_{\mathcal{D}}$, a constrained probabilistic model defines the probability of generating an artefact x_i as

$$P(x_i) \propto \begin{cases} P_{U_{\mathcal{D}}}(x_i) & \text{if } x_i \text{ satisfies } \mathcal{C} \\ 0 & \text{otherwise} \end{cases}$$

By defining constraints explicitly, the model can be trained on artefacts from $U_{\mathcal{D}}$ generally, maintain independence assumptions that maximize expressivity, and ensure probability within the generative model is only assigned to artefacts which belong to \mathcal{D} .

There are several types of constrained probabilistic models including multi-valued decision diagrams (MDDs) for sequential domains (Perez and Régin 2017); MDDs that enforce constraints on non-discretized temporal sequences (Roy et al. 2016); factor graphs for imposing constraints represented as regular languages Papadopoulos et al.; and non-homogeneous Markov models (Pachet, Roy, and Barbieri 2011). Each model incorporates a probabilistic element designed to imitate statistical properties of a corpus—with model parameters (e.g., Markov order or context length) that control the degree of similarity to the corpus—and constraints to guarantee specifiable characteristics of the application domain. Previous work has also shown how constraints can be used avoid plagiarism (i.e., limit the model’s output domain to \mathcal{D} less the artefacts used for training) (Papadopoulos and Roy 2014). It is of interest to note that much of the language used to describe the implementation of these models mirrors closely the language used to by creative theorists to describe the relationship between novelty and typicality. For example, Perez and Régin (2017) describe the process by which the model generates new phrases as a “sampling of the solution set while respecting probabilities,” specifying that the solution set “incorporate[s] some side constraints defining the type of phrases we would like to obtain.”

Quality Assurance

We have discussed how constrained probabilistic models are well-suited for explicitly modeling typicality and novelty, but what about quality? As Boden (2003) puts it, “a computer could merrily produce novel combinations till kingdom come. But would they be of any interest?” How well are constrained probabilistic models able to produce or evaluate *quality*?

To the extent that quality can be represented in either the system’s probabilistic model *and/or* the system’s constraint set, a constrained probabilistic model is naturally endowed with a function for evaluating the quality of the artefacts. By structuring the system’s probabilistic model such that high quality artefacts (by some definition of quality) are assigned higher probability, the system will naturally gravitate towards stochastically generating artefacts of value (as will be shown in our demonstrative example). In cases where quality is a function of the presence or absence of certain characteristics (consider, for example, assessing quality based on the presence of satisfactory rhymes), the system’s constraints can ensure that only artefacts of some minimum quality threshold are generated.

A constrained probabilistic model thus does not define its own function for evaluating quality, but *does* inherently encode one in the forms of probabilistic models and sets of constraints (both of which could be explicitly defined or themselves learned from some training data, as demonstrated in (Bodily, Bay, and Ventura 2017)).

Non-Homogeneous Markov Models

We describe a computational creative system for generating mnemonic devices using a non-homogeneous Markov model (NHMM), a constrained probabilistic model that is also called a constrained Markov model (Pachet, Roy, and Barbieri 2011).

A Markov model \mathcal{M} is a stochastic, probabilistic model defined over a finite state space that strictly adheres to the Markov property, meaning \mathcal{M} is memory-less beyond a finite window. The set of all sequences $s = s_1, \dots, s_n$ of length n generated by \mathcal{M} is represented by S (in our current example this can be thought of as being equivalent to $U_{\mathcal{D}}$ from above). Every sequence $s \in S$ has a non-zero probability equivalent to

$$P_{\mathcal{M}}(s) = P_{\mathcal{M}}(s_1) \cdot P_{\mathcal{M}}(s_2|s_1) \cdots P_{\mathcal{M}}(s_l|s_{n-1})$$

\mathcal{M} is constructed by computing the probability matrix $P_{\mathcal{M}}$ from training examples.

A non-homogeneous Markov model \mathcal{N} is constructed from a Markov model \mathcal{M} , a sequence length l , and a finite sequence of unary constraints $\{C_1, \dots, C_l\}$. The set of solutions for \mathcal{N} is represented by S_C (equivalent to bounded \mathcal{D} from above). With the constraints applied to \mathcal{N} , the probabilities of sequences generated by \mathcal{N} must equal the probability of the same sequence generated by \mathcal{M} :

$$P_{\mathcal{N}}(s) = \begin{cases} P_{\mathcal{M}}(s) & \text{if } s \in S_C \\ 0 & \text{otherwise} \end{cases}$$

\mathcal{N} initially constructs $l - 1$ probability matrices identical to $P_{\mathcal{M}}$ in \mathcal{M} , one for each transition in the sequence to be generated. States or transitions that violate a constraint are removed. Arc consistency is then enforced on the probability matrices, meaning that states or transitions that do not lead to a solution $s \in S_C$ are removed (see Figure 3b). Because the probability matrices in the NHMM are arc consistent and therefore non-zero probabilities are guaranteed to lead to a solution $s \in S_C$. This guarantee of solutions avoids the inefficiency generate-and-check where nearly all samples are rejected when the probability of a solution is small. Finally, the model is re-normalized such that probabilities $P_{\mathcal{N}}(s) = P_{\mathcal{M}}(s|s \in S_C)$ (Pachet, Roy, and Barbieri 2011).

NHMMs have been applied to model music generation, generating melodies constrained to begin and end on the same note (Pachet, Roy, and Barbieri 2011). Barbieri et al. (2012) apply NHMMs to generate lyrics matching rhyme, syllable stress, part-of-speech, and semantic constraints.

NhMMonic

Here we demonstrate the application of constrained probabilistic modeling to computational creativity through *non-homogeneous Markov modeling of mnemonics* (abbreviated as NhMMonic). We define a *mnemonic task* as a sequence of words $s = s_1, \dots, s_l$ to be memorized. A *mnemonic device* then is a sequence of words $m = m_1, \dots, m_l$ of the same length generated such that for all $1 \leq i \leq l$ the first letters in the words s_i and m_i are constrained to be the same (see Figure 3). The primary purpose of a mnemonic device

is to aid in memorization of the order and/or identity of a s by finding a more memorable sequence m that through its constrained similarity to s can serve as a reminder of s . The value of an artefact in this domain is heavily predicated on its effectiveness in facilitating memory.

To our knowledge no mnemonic device generation models have been formally presented. We find that most available Mnemonic generation tools online use what we will call a *template* method. The template method for mnemonic generation first determines a sequence of part of speech constraints as a function of the length l of the sequence to be generated. Words matching these constraints and the aforementioned first-letter constraints are randomly selected from a word bank to fit into the specific sentence structure. The shortcoming to most template-based methods is that they do not model transitions between words, resulting in phrases that exhibit grammatical cohesion, but not semantic cohesion.

Because NHMMs explicitly model transitions between words while allowing for constraints, we consider this model a good candidate for the mnemonic problem. Although NHMMs can and have been used to impose part-of-speech constraints or templates, we chose not to include these constraints in our NHMM implementations preferring to demonstrate that even a relatively simple NHMM can provide good results. While we expect both models to be capable of generating novelty (or *uniqueness* as it is labeled in our survey), we expect NHMMs to outperform other mnemonic device models when it comes to the aspects of typicality relating to grammatical/semantic cohesion and ease of memorization.

Methods

In assessing the NhMMonic system we applied two variants of NHMMs. NHMM-1 has a Markov order of 1 and NHMM-2 has a Markov order of 2 (essentially treating each *pair* of words as a single state token). A higher Markov order allows the mnemonic output to more closely resemble the sample text, increasing the model's cohesion and typicality. A drawback of having a higher Markov order is that fewer solutions $s \in S_C$ are found and in some cases no solutions are found given finite training sentences. NHMM-1, with its lower Markov order, allows our system to find solutions when NHMM-2 does not.

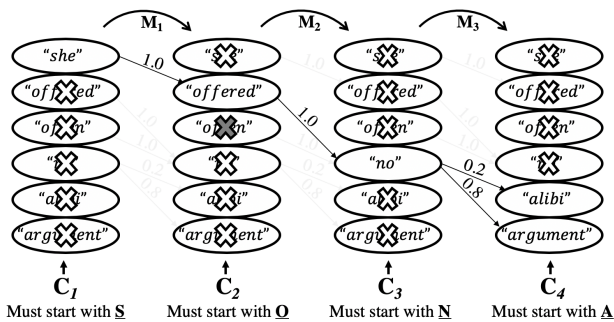
For a mnemonic task $s = s_1, \dots, s_l$, we derive a unary constraint at position i to ensure that the first character of the sequence variable m_i matches the first given letter of s_i . For the purposes of improved readability of generated mnemonics we impose a few additional constraints. For NHMM-1, we constrain each sequence variables m_i to be at least 4 letters long and the last variable m_l to have ended a sentence in the training set. For NHMM-2 the only added constraint is to ensure that the last variable m_l is not a pronoun, preposition, conjunction, or determiner.

The code for the NHMMs used by the NhMMonic system are available in both a C++ implementation² (used for

²<https://github.com/po-gl/ConstrainedMarkovModel>

Stream-enterer, Once-returner, Non-returner, Arahant

(a) A Mnemonic Task



(b) Constrained Probabilistic Model (NHMM)

“She offered no argument”

(c) Mnemonic Device Generation

Figure 3: *The NhmMonic model*. (a) A mnemonic task (i.e., the four stages of enlightenment) to be memorized. (b) A non-homogeneous Markov model built to solve the mnemonic task. M_1 , M_2 , and M_3 represent Markov constraints; C_1 , C_2 , C_3 , and C_4 denote unary constraints derived from the task. Nodes marked with white X’s are removed due to violation of unary constraints while the node marked with a grey X is removed to keep the model arc consistent. Edge labels indicate transition probabilities. (c) A possible mnemonic generated by the model.

NHMM-1) and a Java implementation³ (used for NHMM-2) online

Results

To evaluate the use of constrained Markov models for generating mnemonic devices, we devised an online survey to compare four different mnemonic device generation models:

- **Template**—a third-party model⁴ that selects a part-of-speech template to match the desired sequence length and then randomly selects words matching part of speech and initial letter constraints from a hand-crafted word bank.
- **NHMM-0**—a model which randomly selects words matching initial letter constraints with probability derived from word frequencies in the training corpus.
- **NHMM-1**—a first-order NHMM as described above.
- **NHMM-2**—a second-order NHMM as described above.

The latter three models were trained on the COCA dataset (Davies 2009). NHMM-0 and NHMM-1 were trained on 6.8 million sentences from fictional works written between the

³<https://github.com/norkish/downbythebay/tree/master/DownByTheBay/src/dbtb/markov>

⁴Available via <https://spacefem.com/mnemonics>

years 1995 and 2015 while NHMM-2 trained on 3 million sentences from the same works.

Each model was used to generate 4 mnemonic devices for each of 19 different memorization tasks⁵ (Figure 6 shows some examples of tasks included in the experiment). NHMM-2 was able to find satisfying solutions to 12 of the tasks.

To evaluate the generated mnemonics, we designed a survey in which each evaluation consisted of four parts:

1. The respondent was shown one of the 19 memorization tasks for 10 seconds.
2. The respondent was then shown a mnemonic device for the memorization task for 10 seconds (selected randomly from those generated by the four models).
3. The respondent was then given the (unordered) words from the original memorization task and asked to put them in the correct order based on his/her memory of the task and the mnemonic.
4. Lastly the respondent was asked to evaluate the mnemonic device (using Likert scales from 1 to 5) for
 - (a) *memory*—ease of memorization
 - (b) *flow*—grammatical/semantic coherence
 - (c) *creativity*—overall creative value
 - (d) *uniqueness*—degree of novelty

Each respondent completed four evaluations in this manner.

A total of 80 individuals completed the survey for a total of 320 mnemonic device evaluations. The survey was distributed to different social media websites, such as Reddit, Facebook, and Twitter. No personal information was gathered before or after the survey was taken. Figure 4 shows average scores for the four evaluated characteristics by model. The NHMM-2 model made notable improvements over other models in the categories of ease of memorization (*memory*) and grammatical/semantic cohesion (*flow*). Although the NHMM-0 model performed relatively poorly on *memory*, *flow*, and *creativity*, this model was considered equally capable of generating novelty (i.e., *uniqueness*).

Figure 5 shows the impact of task length on ease of memorization, showing generally that the longer a mnemonic task is, the more difficult mnemonics generated for the task are to remember. The graph also shows, however, that the NHMMs and NHMM-2 in particular, is able to generate mnemonics that maintain ease of memorization even for longer tasks.

Figure 6 shows seven mnemonic device tasks together with the highest-rated mnemonic devices (as per average memory score) generated by NhmMonic for the task.

Discussion

Survey results demonstrate that increased grammatical/semantic cohesion afforded by probabilistic Markov models are associated with gains in ease of memorization.

⁵Mnemonics for all models can be seen at <https://tinyurl.com/yxczj7>

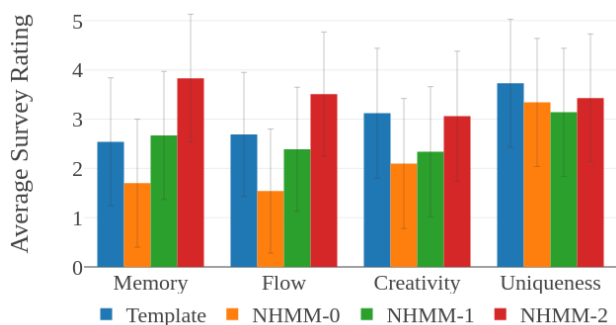


Figure 4: *Survey Results*. Average ratings from 320 evaluations across four metrics for four different mnemonic device generation algorithms. Error bars are standard deviation. The ease of memorization of mnemonics from the NHMM-2 model appears to be associated with improved flow with respect to other models.

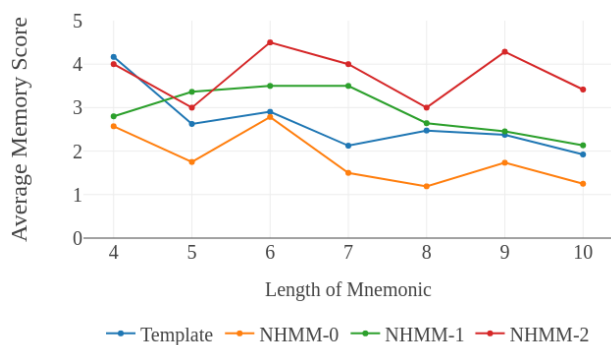


Figure 5: *Impact of Task Length*. As the length of the memorization task increases, the effectiveness of mnemonic devices decreases across all models, but at a much lesser rate for the NHMM-1 and NHMM-2 models. We hypothesize that this is owing to the sustained grammatical and semantic flow that these models achieve from the constrained Markov model.

The fact that increasing the Markov order leads to further gains in both *flow* and *memory* is further evidence of this correlation. These gains from increasing the Markov order were also mirrored in increased creativity scores, suggesting that in the domain of mnemonic device generation, there is an association between the creative success of a mnemonic device and how easily it can be remembered.

This association between the success or popularity of an artefact and the ease with which the brain is able to process and remember it has been observed in creative domains that do *not* deal directly with memorization tasks. A notable example is the study by Nunes, Ordanini, and Valsesia (2014) that demonstrates an association between the popularity of music and the degree of repetition in the song. Researchers observed that increased repetitiveness contributed to higher “processing fluency”, meaning the ease with which the brain is able to grasp a new concept or artefact. A constrained Markov model, through its probabilistic transition model, naturally assigns higher probability to frequent word transitions (which we might assume have higher processing fluency) while using constraints to ensure that generated mnemonics also satisfy the basic requirements of a mnemonic device.

As is typical of Markov-based models, increasing the Markov order can also have negative consequences. The higher the order the more similarity exists between generated artefacts and the training data. Increasing the order also increases the likelihood of the model not being able to find a solution that satisfies both the (now more stringent) Markov constraints and non-Markovian constraints. Both of these problems can be overcome by training on more training data, but the amount of training data needed to sufficiently eradicate the problem increases exponentially with the Markov order.

Independent of the model training, some mnemonic tasks are inherently more difficult owing to the low frequency of words and word beginning with certain letters (this is, of course, language-specific). Consider for example trying to devise a mnemonic device in the English language for the first five dynasties of China, “Neolithic, Xia, Shang, Zhou, Qin”. Solutions certainly exist, but unless the model sees examples in training of word pairs that would be suitable for each word pair in the task (less likely for infrequent collocations), the model will not be able to find them. On these types of tasks we might expect the non-Markovian models to perform better.

We considered other variations of constraints that might have further improved the results of our model. One improvement considered was to constrain more than just the first letter of each word in the mnemonic to match the task. We thought this might further increase the ease of memorization. However, it is generally the case that as constraints become more strict, the model is able to find fewer solutions, often leading to the model being unable to find satisfying solutions. Another improvement we considered was combining the Template and NHMM approaches through part of speech constraints in the NHMM model. We also considered ways to impose semantic themes within mnemonic devices either through unary semantic constraints or through vary-

<u>Four Stages of Enlightenment</u> : Stream-enterer, Once-returner, Non-returner, Arahant “ <i>She offered no argument</i> ”	(NHMM-2, 5.0)
<u>Dantes 9 Circles of Hell</u> : Limbo, Lust, Gluttony, Greed, Anger, Heresy, Violence, Fraud, Treachery “ <i>Lovely little girl giggles as his voice for them</i> ”	(NHMM-1, 5.0)
<u>Last 10 Winners of the FIFA World Cup</u> : France, Germany, Spain, Italy, Brazil, France, Brazil, West Germany, Argentina, Italy “ <i>Four-year-old grandson she is bumped from behind with an inflection</i> ”	(NHMM-2, 4.0)
<u>First 9 ICCA Locations</u> : Lisbon, Mexico City, Dublin, Sydney, Ljubljana, Park City, Paris, Atlanta, Salamanca “ <i>Like most days she looked pretty puny and sickly</i> ”	(NHMM-2, 4.5)
<u>Stages of Grief</u> : Denial, Anger, Bargaining, Depression, Acceptance “ <i>Dreams about being dragged against</i> ”	(NHMM-1, 5.0)
<u>Levels of Biological Organization</u> : Biosphere, Ecosystem, Community, Population, Organism, Organ System, Organ, Tissue, Cell, Molecule “ <i>Blue eyes could pick out one of those clownish men</i> ”	(NHMM-2, 4.5)
<u>Cell Mitosis Cycle</u> : Interphase, Prophase, Prometaphase, Metaphase, Anaphase, Telophase, Cytokinesis “ <i>I pushed past me and the career</i> ”	(NHMM-2, 5.0)

Figure 6: *Top-rated mnemonics generated by NhMMonic*. Seven mnemonic device tasks are shown. Each task consists of a description (bold and underlined) followed by a list of words requiring a mnemonic device. Below each task is the NhMMonic-generated mnemonic device that received the highest memorization score (with the exact model and score given in parentheses).

ing the training data. We leave these as exploratory ideas for future work.

Many forms of creativity have relational structure (e.g., rhyme schemes, repeated motifs, etc.). Unlike the example we have shown here which uses solely unary constraints, relational structure is most effectively realized using binary constraints. Sampling from constrained Markov models with binary constraints is known to be a much harder problem (see (Rivaud and Pachet 2017)), however recent work has been done towards providing reasonable solutions (Papadopoulos et al. 2015; Roy et al. 2016). This has relevance for imposing semantic constraints in models of mnemonic device generation because binary constraints can effectively be used to impose *floating constraints* (i.e., constraints that can be satisfied at variable positions) rather than specifying a specific word position where semantic constraints must be satisfied.

NHMM doesn’t directly model all aspects of creativity. For example intention, explicit self-evaluation, others? Constraints themselves can be learned or imitated. One ramification of learned constraints is that in addition to whatever constraints are required to define typicality, additional constraints could themselves be probabilistically applied in generating artefacts. This would allow constraints to be “broken” (or rather never applied) with some degree of probability, demonstrating a method by which rules can be “intelligently” broken.

In this work we have discussed aspects of constrained probabilistic modeling that are well-suited for consistently generating novelty and typicality in computational creative artefacts. As an example, we have demonstrated the application of non-homogeneous Markov models to the problem of mnemonic device generation. Our results suggest that the constrained Markov model approach is able to effectively generate mnemonic devices that satisfy basic requirements of mnemonic devices while exhibiting elevated

levels of grammatical/semantic flow, ease of memorization, and creative value.

Acknowledgements

Many thanks to the team at spacefem.com for their assistance using the spacefem mnemonic device generator.

References

- Barbieri, G.; Pachet, F.; Roy, P.; and Esposti, M. D. 2012. Markov constraints for generating lyrics with style. In *Proceedings of the Twentieth European Conference on Artificial Intelligence*, 115–120.
- Boden, M. A. 2003. *The Creative Mind: Myths and Mechanisms, Second Edition*. Routledge.
- Bodily, P.; Bay, B.; and Ventura, D. 2017. Computational creativity via human-level concept learning. In *Proceedings of the Eighth International Conference on Computational Creativity*, 57–64.
- Colton, S., and Wiggins, G. A. 2012. Computational creativity: The final frontier? In *Proceedings of the Twentieth European Conference on Artificial Intelligence*, 21–26. IOS Press.
- Csikszentmihályi, M. 1996. *Flow and the Psychology of Discovery and Invention*. Harper Perennial.
- Davies, M. 2009. The 385+ million word Corpus of Contemporary American English (1990–2008+): Design, architecture, and linguistic insights. *International Journal of Corpus Linguistics* 14(2):159–190.
- Jordanous, A. 2016. Four PPPPerspectives on computational creativity in theory and in practice. *Connection Science* 28(2):194–216.
- Koren, L. 2010. *Which “aesthetics” do you mean? : Ten definitions*. Point Reyes, California: Imperfect Publishing.

- Morris, R. G.; Burton, S. H.; Bodily, P. M.; and Ventura, D. 2012. Soup Over Bean of Pure Joy : Culinary ruminations of an artificial chef. In *Proceedings of the Third International Conference on Computational Creativity*, 119–125.
- Nunes, J. C.; Ordanini, A.; and Valsesia, F. 2014. The power of repetition: Repetitive lyrics in a song increase processing fluency and drive market success. *Journal of Consumer Psychology* 25(2):187–199.
- Onarheim, B., and Biskjaer, M. M. 2017. Balancing constraints and the sweet spot as coming topics for creativity research. In *Creativity in design: Understanding, capturing, supporting*. APA.
- Pachet, F.; Roy, P.; and Barbieri, G. 2011. Finite-length Markov processes with constraints. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*.
- Papadopoulos, A., and Roy, P. 2014. Avoiding plagiarism in Markov sequence generation. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2731–2737.
- Papadopoulos, A.; Pachet, F.; Roy, P.; and Sakellariou, J. 2015. Exact sampling for regular and Markov constraints with belief propagation. In *Proceedings of the International Conference on Principles and Practice of Constraint Programming*, 341–350. Springer.
- Pease, A.; Guhe, M.; and Smaill, A. 2010. Some aspects of analogical reasoning in mathematical creativity. In *Proceedings of the First International Conference on Computational Creativity*, 60–64.
- Perez, G., and Régin, J.-C. 2017. MDDs: Sampling and probability constraints. In *Proceedings of the Twenty-Third International Conference on Principles and Practice of Constraint Programming*, 226–242. Springer, Cham.
- Pérez y Pérez, R., and Sharples, M. 2004. Three computer-based models of storytelling: BRUTUS, MINSTREL and MEXICA. *Knowledge-Based Systems*.
- Ritchie, G. 2007. Some empirical criteria for attributing creativity to a computer program. *Minds and Machines* 17(1):67–99.
- Rivaud, S., and Pachet, F. 2017. Sampling Markov models under constraints: Complexity results for binary equalities and grammar membership. *arXiv preprint*.
- Roy, P.; Perez, G.; Régin, J.-C.; Papadopoulos, A.; Pachet, F.; and Marchini, M. 2016. Enforcing Structure on Temporal Sequences: The Allen Constraint. In *Proceedings of the International Conference on Principles and Practice of Constraint Programming*, 786–801. Springer.
- Saunders, R., and Gero, J. S. 2001. The Digital Clockwork Muse: A Computational Model of Aesthetic Evolution. In *Proceedings of the Artificial Intelligence and Simulation of Behavior Convention*, 12–21.
- Ventura, D. 2017. How to Build a CC System. In *Proceedings of the Eighth International Conference on Computational Creativity*, 253–260.
- Wiggins, G. A. 2006. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems* 19(7):449–458.

NONOTO: A Model-agnostic Web Interface for Interactive Music Composition by Inpainting

Théis Bazin

Sony CSL
Paris, France
theis.bazin@outlook.com

Gaëtan Hadjeres

Sony CSL
Paris, France
gaetan.hadjeres@sony.com

Abstract

Inpainting-based generative modeling allows for stimulating human-machine interactions by letting users perform stylistically coherent local editions to an object using a statistical model. We present *NONOTO*, a new interface for interactive music generation based on inpainting models. It is aimed both at researchers, by offering a simple and flexible API allowing them to connect their own models with the interface, and at musicians by providing industry-standard features such as audio playback, real-time MIDI output and straightforward synchronization with DAWs using Ableton Link.

Keywords interfaces, generative models, inpainting, interactive music generation, web technologies, open-source software

Introduction

We present a web-based interface that allows users to compose symbolic music in an interactive way using generative models for music. We strongly believe that such models only reveal their potential when actually used by artists and creators. While generative models for music have been around for a while (Boulangier-Lewandowski, Bengio, and Vincent 2012; Hadjeres, Pachet, and Nielsen 2017; Roberts et al. 2018), the conception of A.I.-based interactive interfaces designed for music creators is still burgeoning. We contribute to this emerging area by providing a general web interface for many music generation models so that researchers in the domain can easily test and promote their works in actual music production and performance settings. This desire follows from the seminal work by Theis, van den Oord, and Bethge, in which the authors advocate that quantitative evaluation of generative models in an unambiguous way is hard and that "generative models need to be evaluated directly with respect to the application(s) they were intended for" (Theis, van den Oord, and Bethge 2015). Lastly, we hope that the present work will contribute in making A.I.-assisted composition accessible to a wider audience, from non musicians to professional musicians, helping bridge the gap between these communities.

Drawing inspiration from recent advances in interactive interfaces for image restoration and editing (Isola et al. 2016; Jo and Park 2019; Yu et al. 2018), we focus on provid-

ing an interface for "inpainting" models for symbolic music, which are models that are able to recompose *a portion of a score* given all the other portions. The reason is that such models are more suited for an *interactive* use (compared to models generating a full score all at once) and let users play an active part in the compositional process. As an outcome, users can feel that the composition is the result of their work and not just something created by the machine. Furthermore, allowing quick exploration of musical ideas in a playful setting can enhance creativity and provide accessibility: the "technical part" of the composition is taken care of by the generative model which allows musicians as well as non experts in music to express themselves more freely.

Contributions: The key elements of novelty are: (a) easy-to-use and intuitive interface for users, (b) easy-to-plug interface for researchers allowing them to explore the potential of their music generation algorithms, (c) web-based and model-agnostic framework, (d) integration of existing music inpainting algorithms, (e) novel paradigms for A.I.-assisted music composition and live performance, (f) integration in professional music production environments.

The code for the interface is distributed under a GNU GPL license and available, along with ready-to-use packaged standalone applications and video demonstrations of the interface, on our GitHub¹.

Existing approaches

The proposed system is akin to the FlowComposer system (Papadopoulos, Roy, and Pachet 2016) which offers to generate sheets of music by performing local updates (using Markov Models in their case). However, this interface does not exhibit the same level of interactivity as ours since no real-time audio nor MIDI playback is available, which limits the tool to solely studio usage and makes for a less spontaneous and reactive user experience.

The recent tools proposed by the Google Magenta team as part of their Magenta Studio effort (Adam Roberts 2019) are more aligned with our aims in this project: they offer a selection of Ableton Live plugins (using Max for Live) that make use of various symbolic music generation models for rhythm as well as melody (Roberts et al. 2018;

¹<https://github.com/SonyCSLParis/NONOTO>

Huang et al. 2018). Similarly, the StyleMachine, developed by Metacreative Technologies ², is a proprietary tool that allows to generate midi-tracks into Ableton Live in various dance music styles using a statistical model trained on different stylistic corpora of electronic music. Yet these tools differ from ours in the generation paradigm used: they offer either continuation-based (the model generates the end of a sequence given the beginning) or complete generation (the model generates a full sequence, possibly given a template), thus breaking the flow of music on new generations. We believe that this limits their level of interactivity as opposed to local, inpainting-based models as ours, as mentioned previously. In particular, it hinders their usage in live, performance contexts.

Suggested mode of usage

The interface displays a musical score which loops forever as shown in Figure 1. Users can then modify the score by regenerating any region only by clicking/touching it. The displayed musical score is updated instantly without interrupting the music playback. Other means of control are available depending on the specificity of the training dataset: we implemented, for instance, the positioning of fermatas in the context of Bach chorales generation or the control of the chord progression when generating Jazz leadsheets or Folk songs. These *metadata* are sent, along with the sheet, to the generative models when performing re-generations. The scores can be seamlessly integrated in a DAW so that the user (or even other users) can shape the sounds, add effects, play the drums or create live mixes. This creates a new jam-like experience in which the user controlling the A.I. can be seen as just one of the multiple instrument players. This interface thus has the potential to create a new environment for collaborative music production and performance.

Since our approach is flexible, our tool can be used in conjunction with other A.I.-assisted musical tools like Magenta Studio (Adam Roberts 2019) or the StyleMachine (from Metacreative Technologies).

Technology

Framework

Our framework relies on two elements: an interactive web interface and a music inpainting server. This decoupling is strict so that researchers can easily plug-in new inpainting models with little overhead: it suffices to implement how the music inpainting model should function given a particular user input. We make heavy use of modern web browser technologies, making for a modular and hackable code-base for artists and researchers, allowing e.g. to edit the interface to allow for some particular means of interaction or to add support for some new metadata specific to a given corpus.

Interface The interface is based on OpenSheetMusicDisplay ³, a TypeScript library aimed at rendering MusicXML

²<https://metacreativetech.com/products/stylemachine-lite/>

³<https://github.com/opensheetmusicdisplay/opensheetmusicdisplay/>

sheets with vector graphics. Using Tone.js (Mann 2015), a JavaScript library for real-time audio synthesis, we augmented OSMD with real-time audio playback capacities, allowing users to preview the generated music in real-time from within the interface. Furthermore, the audio playback is uninterrupted by re-generations, enabling a truly interactive experience.

Generation back-end and communication For better interoperability, we rely on the MusicXML standard to communicate scores between the interface and the server. The HTTP-based communication API then just consists in two commands that are required server-side:

- A `/generate` command which expects the generation model to return a fresh new sheet of music in the MusicXML format to initialize a session,
- A `/timerange-change` command which takes as parameter the position of the interval to re-generate. The server is then expected to return an updated sheet with the chosen portion regenerated by the model using the current musical context.

DAW integration In order for NONOTO to be readily usable in traditional music production and performance contexts, we implemented the possibility of integrating the generated scores in any DAW in real time. To this end, we provide the user with the option of either rendering the generated sheet to audio in real-time from within the web interface using Tone.js or of routing it via MIDI to any virtual MIDI port on the host machine, using the JavaScript bindings to the Web MIDI API, WebMidi.js ⁴. We also integrated support for Ableton Link ^{5,6}, an open-source technology developed by Ableton for easy synchronization of musical hosts on a local network, allowing to synchronize the interface with e.g. Ableton Live. Adding support for these technologies does not represent novel advances on our side *per se*, yet, paired with the support of arbitrary generation back-ends, they allow to quickly test new generation models in a standard music production environment with minimal overwork and make for a beneficial tool for researchers – and the first of its kind to our knowledge.

Conclusion

We have introduced NONOTO, an interactive, open-source and hackable interface for music generation using inpainting models. We invite researchers and artists alike to make it their own by developing new models or means of interacting with those. This high level of hackability is to a large extent permitted by the wide range of technologies now offered in a very convenient fashion by modern web browsers, from which we draw heavily. Ultimately, we hope that providing tools such as ours with a strong focus on usability, accessibility, affordance and hackability will help shift the general perspective on machine learning for music creation, transitioning from the current and somewhat negative view

⁴<https://github.com/djipco/webmidi>

⁵<https://github.com/Ableton/link/>

⁶<https://github.com/2bbb/node-abletonlink>

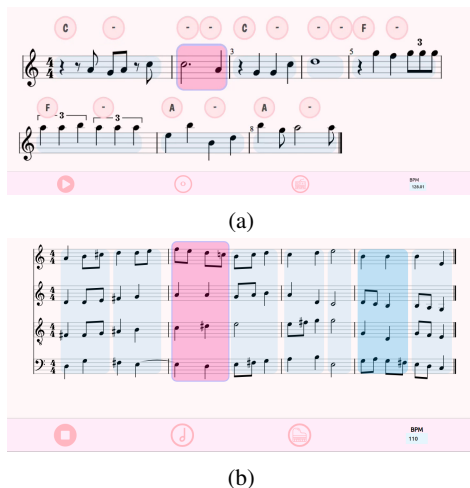


Figure 1: Our web interface used on different datasets: 1a melody and symbolic chords format, 1b four-part chorale music.

of "robot music", replacing musicians, towards a more realistic and humbler view of it as A.I.-assisted music.

Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful comments.

References

- Adam Roberts, Yotam Mann, J. E. C. R. 2019. Magenta studio. <https://magenta.tensorflow.org/studio-announce>.
- Boulanger-Lewandowski, N.; Bengio, Y.; and Vincent, P. 2012. Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription. *arXiv e-prints* arXiv:1206.6392.
- Hadjeres, G.; Pachet, F.; and Nielsen, F. 2017. DeepBach: a steerable model for Bach chorales generation. In *Proc. of the 34th International Conference on Machine Learning (ICML)*, 1362–1371.
- Huang, C. A.; Vaswani, A.; Uszkoreit, J.; Shazeer, N.; Hawthorne, C.; Dai, A. M.; Hoffman, M. D.; and Eck, D. 2018. An improved relative self-attention mechanism for transformer with application to music generation. *CoRR* abs/1809.04281.
- Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2016. Image-to-image translation with conditional adversarial networks. *ArXiv*.
- Jo, Y., and Park, J. 2019. SC-FEGAN: Face Editing Generative Adversarial Network with User's Sketch and Color. *arXiv e-prints* arXiv:1902.06838.
- Mann, Y. 2015. Interactive music with Tone.js. In *Proceedings of the 1st Web Audio Conference*.
- Papadopoulos, A.; Roy, P.; and Pachet, F. 2016. Assisted lead sheet composition using FlowComposer. In Rueher, M.,

ed., *Principles and Practice of Constraint Programming*, 769–785. Cham: Springer International Publishing.

Roberts, A.; Engel, J.; Raffel, C.; Hawthorne, C.; and Eck, D. 2018. A hierarchical latent vector model for learning long-term structure in music. In *Proceedings of the 35th International Conference on Machine Learning*, 4364–4373.

Theis, L.; van den Oord, A.; and Bethge, M. 2015. A note on the evaluation of generative models. *arXiv e-prints* arXiv:1511.01844.

Yu, J.; Lin, Z.; Yang, J.; Shen, X.; Lu, X.; and Huang, T. S. 2018. Free-form image inpainting with gated convolution. *CoRR* abs/1806.03589.

Neural Drum Machine : An Interactive System for Real-time Synthesis of Drum Sounds

Cyran Aouameur

Sony CSL Paris

cyran.aouameur@sony.com

Philippe Esling

IRCAM

esling@ircam.fr

Gaetan Hadjeres

Sony CSL Paris

gaetan.hadjeres@sony.com

Abstract

In this work, we introduce a system for real-time generation of drum sounds. This system is composed of two parts: a generative model for drum sounds together with a Max4Live plugin providing intuitive controls on the generative process. The generative model consists of a Conditional Wasserstein autoencoder (CWAE), which learns to generate Mel-scaled magnitude spectrograms of short percussion samples, coupled with a Multi-Head Convolutional Neural Network (MCNN) which estimates the corresponding audio signal from the magnitude spectrogram. The design of this model makes it lightweight, so that it allows one to perform real-time generation of novel drum sounds on an average CPU, removing the need for the users to possess dedicated hardware in order to use this system. We then present our Max4Live interface designed to interact with this generative model. With this setup, the system can be easily integrated into a studio-production environment and enhance the creative process. Finally, we discuss the advantages of our system and how the interaction of music producers with such tools could change the way drum tracks are composed.

Introduction

In the early '80s, the widespread use of the sampler revolutionized the way music is produced: besides hiring professional musicians, music producers have since been able to compose with sampled sounds. This has brought much flexibility for both drum and melody production, thanks to the various offline edition possibilities offered by such systems like pitch shifting, time stretching, looping and others.

Nowadays, many producers still rely on samplers for drums production, mainly due to the always-increasing amount of samples libraries available for download. This has helped music production become increasingly accessible, even to newcomers with no or little notion in sound design. However, relying on samples has also some drawbacks. Indeed, producers now have to browse their vast collection of samples in order to find the "right sound". This process is often inefficient and time-consuming. Kick drum datasets are usually unorganized with, for instance, samples gathered in a single folder, regardless of whether they sound "bright" or "dark". As a result, many producers would rely

only on a limited selection of their favourite sounds, which could hamper creativity.

Hence, a method allowing a comfortable and rich exploration of sounds becomes an essential requirement in music production, especially for non-expert users. Numerous research efforts have been done in the domain of user experience in order to provide interfaces that enhance the fluidity of human-machine interactions. As an example, synthesizers interfaces now often feature "macro" controls that allow to tune a sound to one's will quickly.

Another approach to tackle this problem is the use of Music Information Retrieval (MIR) to deal more efficiently with vast libraries of audio samples. MIR is an approach based on feature extraction: by computing a lot of audio features (Peeters 2004) over a dataset, one can define a perceptual similarity measure between sounds. Indeed, audio features are related to perceptual characteristics, and a distance between a combination of features is more relevant than a squared error between two waveforms. The combination of MIR with machine learning techniques appears natural in order to organize such audio libraries by allowing, for example, clustering or classification based on audio content. We can cite software such as AudioHelper's Samplism, Sononym and Algonaut's Atlas.

While such software only allows one to organize an existing database, we propose to use artificial intelligence to intuitively generate sounds, thus also tackling the problem of sound exploration. Only very recently, some machine learning models have been developed specifically for the problem of audio generation. These *generative models* perform what we could define as *synthesis by learning*. They rely on generative modelling, which allows performing audio synthesis by learning while tackling the question of intuitive parameter control (Esling, Bitton, and others 2018; Engel et al. 2017).

Generative models are a flourishing class of machine learning approaches whose purpose is to generate novel data based on the observation of existing examples (Bishop and Mitchell 2014). The learning process consists of modelling the underlying (and unknown) probability distribution of the data based on samples. Once the model is trained, it is then possible for a user to generate new samples at will. However, for the user to be active during the synthesis process and not only passively browsing the outputs of the system, we find

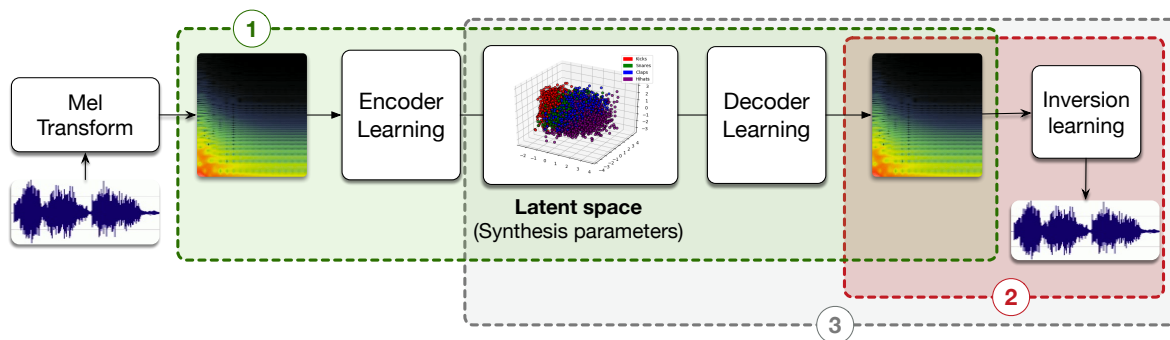


Figure 1: This diagram presents our end-to-end system for drum sounds synthesis. The generative model (1) learns how to reconstruct spectrograms from a parameters' space. Then, the second part of the system (2) is dedicated to spectrogram inversion, to generate some signal from a Mel spectrogram. Finally, the software interface (3) allows a user to interact with the model and to generate sound from the parameters' space.

crucial the requirement that the system should provide intuitive controls. To this end, we need a model that extracts a compact high-level representation of the data. Then, by providing these simple high-level controls to a user, the synthesis process can be guided by perceptual characteristics. A user would just have to explore a continuous and well-organized parameter space to synthesize an infinite variety of sounds.

Our proposal

In this work, we describe a system that allows to create a controllable audio synthesis space so that we can use it to synthesize novel sounds in an intuitive manner. This system can be split into three components (Fig. 1):

- A Conditional Wasserstein Auto-Encoder (CWAE) which generates Mel-scaled spectrograms.
- An extension of the Multi-Head Convolutional Neural Network (MCNN) which reconstructs signal from Mel-scaled spectrograms.
- A Max4Live plugin allowing users to interact with the model in a music production environment.

In the remainder of this document, we first provide a state of the art on Wasserstein auto-encoders and MCNN. Then we describe our model and the data we used to train it. We discuss reconstruction and generation results. Finally, we showcase the associated plugin and explain how it could change the way drum tracks are produced.

Related work

Generative models on audio waveforms

A few systems based on generative models have been recently proposed to address the learning of latent spaces for audio data. The Wavenet auto-encoder (Engel et al. 2017) combines Wavenet (Oord et al. 2016) with auto-encoders and uses dilated convolutions to learn waveforms of musical instruments. By conditioning the generation on the pitch, such a system is capable of synthesizing musical notes with various timbres. The WaveGAN (Donahue, McAuley,

and Puckette 2018) uses Generative Adversarial Networks (GANs) to generate drum sounds or bird vocalizations by directly learning on waveform. However, the GAN approach provides little control over the generation because it is still difficult to structure their latent space.

Generative models on spectral representations

Other works have focused on generating sound as spectrograms, a complex time-frequency representation of sound. This visual representation of sound intensity through time allows us to treat sounds like images, but has to reverted back to the signal domain to produce sound. In (Esling, Bitton, and others 2018) uses VAEs to learn a generative space where instrumental sounds are organized with respect to their timbre. However, because the model is trained on spectra frames, it lacks temporal modeling. This hampers the capacity of the model to easily allow users to generate evolving structured temporal sequences such as drum sounds. This approach introduced in (Donahue, McAuley, and Puckette 2018) takes into account these temporal dependencies by proposing SpecGAN, a generative models which uses GANs to generate spectrograms as if they were images.

Spectrogram inversion

Working with neural networks often forces us to discard the phase information of a spectrogram. Therefore, one cannot use the inverse Fourier transform to retrieve the signal it originates from. With classic STFT, a common workaround is to use the Griffin-Lim Algorithm (GLA) (Griffin and Lim 1984) which allows to estimate the missing phase information. Also, The Multi-head Convolutional Neural Network (MCNN) is a model that inverts STFTs (Arik, Jun, and Diamos 2019) using neural networks.

However, STFT is not the best transform for our purpose. Indeed, Mel-scaled spectrograms are known to be more suitable for training convolutional neural networks (Huzaifah 2017). Mel-scaled spectrograms are computed with filters based on the Mel scale, a perceptual frequency scale that tries to mimic the human perception of pitches.

Despite being more adapted for training, using Mel-scaled spectrograms introduces a problem: they are not invertible and GLA cannot be used. Therefore, some deep learning based models have been developed in order to estimate signal from non-invertible spectrograms. In (Prenger, Valle, and Catanzaro 2018), the authors present WaveGlow, a flow-based network capable of generating high quality speech from Mel spectrograms. Also, in (Huang et al. 2018), the authors use a conditioned Wavenet to estimate signal from Constant-Q Transforms, another non-invertible transform.

Proposed model

Our model is composed of two components: a generative model on spectrograms, whose role is to learn a latent space from our dataset and to generate meaningful spectrograms from this space, and a spectrogram inversion model, whose role is reconstruct waveforms from our generated spectrograms.

Preliminaries on variational autoencoders

To formalize our problem, we rely on a set of data $\{\mathbf{x}_n\}_{n \in [1, N]}$ lying in a high-dimensional space $\mathbf{x}_i \in R^{d_x}$. We assume that these examples follow an underlying probability distribution $p(\mathbf{x})$ that is unknown. Our goal is to train a generative model able to sample from this distribution.

We consider a parametrized *latent variable model*

$$p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z})\pi(\mathbf{z}).$$

by introducing latent variables $\mathbf{z} \in R^{d_z}$ lying in a space of smaller dimensionality than x ($d_z \ll d_x$) and distributed according to the prior $\pi(\mathbf{z})$. We are interested in finding the parameter θ that maximizes the likelihood $\sum_i p_\theta(x_i)$ of the dataset. However, for usual choices of the conditional probability distributions $p_\theta(x|z)$ (typically a deep neural network), this quantity is intractable.

The variational autoencoder (VAE) (Kingma and Welling 2013) is a model that introduces a variational approximation $q_\phi(\mathbf{z}|\mathbf{x})$ to the intractable posterior $p_\theta(\mathbf{x}|\mathbf{z})$ (the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ is often chosen as a parametrized family of diagonal Gaussian distributions). The network $q_\phi(z|x)$ is called the *encoder* whose aim is to produce latent codes given x while the network $p_\theta(x|z)$ is called a *decoder*, which tries to reconstruct x given a latent code z .

The introduction of the variational approximation of the posterior allows us to obtain the following lower bound $\mathcal{L}(\theta, \phi)$ (called ELBO for Evidence Lower Bound) over the intractable likelihood:

$$\mathcal{L}(\theta, \phi) = E_{\mathbf{x} \sim p(\mathbf{x})} \left[\underbrace{E_{\mathbf{z} \sim p(\mathbf{z}|\mathbf{x})} [\log p_\theta(\mathbf{x}|\mathbf{z})]}_{\text{reconstruction}} - \underbrace{D_{\text{KL}}[q_\phi(\mathbf{z}|\mathbf{x}) \parallel \pi(\mathbf{z})]}_{\text{regularization}} \right], \quad (1)$$

where D_{KL} denotes the Kullback-Leibler divergence (Cover and Thomas 2012).

- The first term $E_{\mathbf{z} \sim p(\mathbf{z}|\mathbf{x})} [\log p_\theta(\mathbf{x}|\mathbf{z})]$ is the likelihood of the data \mathbf{x} generated from the set of latent variable $\mathbf{z} \sim q_\phi(z|x)$ coming from the approximate posterior. Maximizing this quantity can be seen as minimizing a *reconstruction error*.
- The second term $D_{\text{KL}}[q_\phi(\mathbf{z}|\mathbf{x}) \parallel \pi(\mathbf{z})]$ is the distance between $q_\phi(\mathbf{z}|\mathbf{x})$ and $\pi(\mathbf{z})$ and can be interpreted as a regularization term.

In (Sohn, Lee, and Yan 2015), the authors add a conditioning mechanism to the original VAE which consists in conditioning all three networks $p_\theta(x|z)$, $q_\phi(z|x)$ and $\pi(z)$ on some metadata m (in most cases, the prior $\pi(z)$ does not depend on m).

However, a known problem of VAEs is that they tend to generate blurry samples and reconstructions (Chen et al. 2016). This becomes a major hindrance in the context of spectrogram reconstructions. Hopefully, this problem can be overcome by the use of Wasserstein Auto-Encoders (WAEs) instead of VAEs. The main difference consists in replacing the D_{KL} term in (1) by another divergence between the prior π and the *aggregated posterior* $q_z(\mathbf{z}) := E_{x \sim p_x} [q(\mathbf{z}|\mathbf{x})]$. In particular, the MMD-WAE considers a Maximum Mean Discrepancy (MMD) (Berlinet and Thomas-Agnan 2011) distance defined as follows:

$$\text{MMD}_k^2(p, q) = \left\| \int k(z, \cdot) p(z) dz - \int k(z, \cdot) q(z) dz \right\|_{\mathcal{H}_k}^2, \quad (2)$$

where $k : \mathcal{Z} \times \mathcal{Z} \rightarrow \mathbf{R}$ is an positive-definite reproducing kernel and \mathcal{H}_k the associated Reproducing Kernel Hilbert Space (RKHS) (Berlinet and Thomas-Agnan 2011). MMD is known to perform well when matching high-dimensional standard normal distributions (Tolstikhin et al. 2017; Gretton et al. 2012). Since the MMD distance is not available in closed form, we use the following unbiased U-statistic estimator (Gretton et al. 2012) for a batch size n and a kernel k :

$$\text{MMD}_{k,n}^2(\pi, q_z) := \frac{1}{n(n-1)} \sum_{l \neq j} k(z_l, z_j) + \frac{1}{n(n-1)} \sum_{l \neq j} k(\tilde{z}_l, \tilde{z}_j) - \frac{2}{n^2} \sum_{l,j} k(z_l, \tilde{z}_j), \quad (3)$$

with $\tilde{z} := \{\tilde{z}_1, \dots, \tilde{z}_n\}$ where $\tilde{z}_i \sim \pi$ and $z := \{z_1, \dots, z_n\}$ where $z_i \sim q_z$.

The Conditional WAE

We now introduce a Conditional WAE (CWAE) architecture so that we can generate spectrograms depending on additional metadata such as the category of the original sound (e.g. kick drum, snare, clap, etc.).

Our encoder is defined as a Convolutional Neural Network (CNN) with l layers of processing. Each layer is a 2-dimensional convolution followed by conditional batch normalization (Perez et al. 2017; Perez et al. 2018) and a ReLU activation. This CNN block is followed by Fully-Connected (FC) layers, in order to map the convolution layers activation to a vector of size d_z which is that of the latent space.

The decoder network is defined as a mirror to the encoder, so that they have a similar capacity. Therefore, we move the FC block before the convolutional one and change the convolution to a convolution-transpose operation. Also, we slightly adjust the convolution parameters so that the output size matches that of the input.

Our convolutional blocks are made of 3 layers each, with a kernel size of (11,5), a stride of (3,2) and a padding of (5,2). Our FC blocks are made of 3 layers with sizes 1024, 512 and $d_z = 64$. Therefore, our latent space is of size $d_z = 64$.

In the case of WAEs, the MMD is computed between the prior π and the aggregated posterior $q_Z(\mathbf{z}) := E_{\mathbf{x} \sim p_X} [q(\mathbf{z}|\mathbf{x})]$. As a result, the latent spaces obtained with WAEs are often really Gaussian which makes them easy to sample. Here, the conditioning mechanism implies that we use separated gaussian priors $\pi_c = \mathcal{N}(0, 1)$ for each class c , in order to be able to sample all classes as Gaussian. Indeed, computing a MMD loss over all classes would force the global aggregated posterior to match the gaussian prior, and thus restrict the freedom for latent positions. Therefore, we have to compute the per-class MMD to backpropagate on.

Let's formalize this problem by decomposing our dataset D into C subsets D_c with $1 \leq c \leq C$, containing all elements from a single class. We define $q_z^c(\mathbf{z}) := E_{\mathbf{x} \in D_c} [q(\mathbf{z}|\mathbf{x}, m = c)]$. Thus, our regularizer is computed as follows :

$$\mathcal{D}_Z(\pi_c, q_z) = \frac{1}{C} \sum_{c=1}^C \text{MMD}_{k,n}^2(\pi, q_z^c). \quad (4)$$

Finally, our loss function is computed as:

$$\mathcal{L}(\theta, \phi) = \sum_{i=1}^n \text{MSE}(x_i, \hat{x}_i) + \beta \mathcal{D}_Z(\pi, q_z), \quad (5)$$

where $\beta = 10$ and k is the *multi-quadratics kernel* as for CelebA in (Tolstikhin et al. 2017).

MCNN inversion

To invert our Mel-spectrograms back to the signal domain, we use a modified version of the original MCNN. In this section, we first review the original MCNN before detailing how we adapted it to handle Mel-spectrograms of drum samples.

MCNN is composed of multiple heads that process STFTs (Fig. 2). These heads are composed of L processing layers combining 1D transposed convolutions and Exponential Linear Units (ELUs). The convolution layers are defined by a set of parameters (f, s, c) , respectively the filter width, the stride and the number of output channels. We multiply the output of every head with a trainable scalar w_i to weight these outputs, and we compute the final waveform as their sum. Lastly, we scale the waveform with a non-linearity (scaled softsign). The model is trained to estimate a waveform which spectrogram matches the original one. For more implementation details, we refer the interested readers to the original article.

We have chosen to use this model because of three main points. First, it performs a fast (300x real-time) and precise

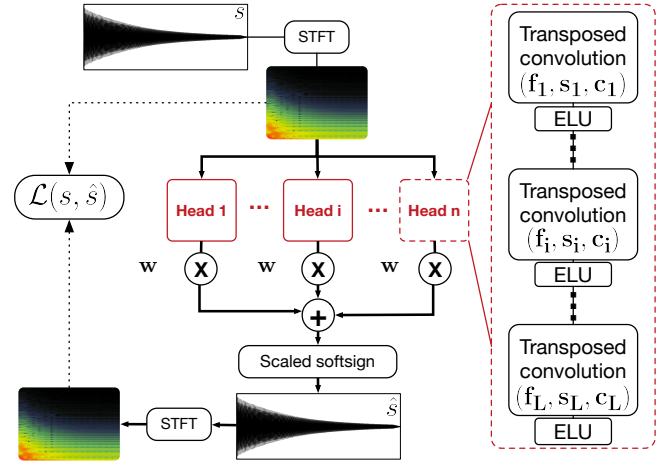


Figure 2: The MCNN for spectrogram inversion. Its multiple heads estimate waveforms that are summed to produce the final waveform. Finally, the loss is computed between the resulting spectrogram and the original one

estimation of a signal given a spectrogram. Then, it can deal with non-invertible transforms that derive from STFT such as Mel-STFT. Finally, its feed-forward architecture allows to takes advantage of GPUs, unlike iterative or auto-regressive models.

In our implementation, we kept most of the parameters suggested in (Arik, Jun, and Diamos 2019). We use a MCNN with 8 heads of $L = 8$ layers each where, for each layer l_i , $1 \leq i \leq L$, we have $(w_i, s_i) = (13, 2)$. However, because we have padded our signals with zeros to standardize their length, two problems appear. First, we observed that the part of the spectrogram corresponding to the padding (made of zeros) was not well reconstructed if convolution feature biases. Without biases, zeros stay zeros throughout the kernel multiplications. Therefore, we removed all biases. Then, we observed a leakage phenomenon: because the convolution filters are quite large (length 13), the reconstructed waveform had more non-zero values than the original one. Therefore, the loss is lower-bounded by this effect. To tackle this problem, we decided to apply a mask to the final output of our model, aiming at correcting this leakage. Thus, for the number of output channels for layer i , we have :

$$c_i = \begin{cases} 2^{L-i} & \text{if } 2 \leq i \leq L \\ 2 & \text{if } i = 1. \end{cases}$$

The output of head h is a couple of 2 vectors (s_h, m_h) . We estimate the mask \hat{M} as follows:

$$\hat{M} = \sigma \left(\sum_{h=1}^8 m_h \right). \quad (6)$$

The finally output waveform \hat{s} is computed as :

$$\hat{s}^* = \sum_{h=1}^8 w_h * s_h, \quad (7)$$

$$\hat{s} = \hat{s}^* \times \hat{M}. \quad (8)$$

To train the mask, we use supervised training and introduce a loss term between the original mask M and the estimated one \hat{M} , that we name *mask loss*:

$$L_{mask}(M, \hat{M}) = \text{BCE}(M, \hat{M}). \quad (9)$$

At generation time the mask is binarized. This solution has worked very well to cut the tail artifacts introduced by the convolutions.

A second change is that we now train MCNN on Mel-scaled spectrograms rather than STFT. However, original losses were computed on STFT. To turn a STFT into a Mel-scaled spectrogram, we compute a filterbank matrix F to combine the 2048 FFT bins into 512 Mel-frequency bins. Finally, we multiply this matrix with the STFT to retrieve a Mel-scaled spectrogram:

$$\text{Mel} = \text{STFT} \times F. \quad (10)$$

Therefore, we can simply convert all STFTs to Mel-scaled spectrograms before the loss computation. This does not affect the training procedure: back-propagation remains possible since this conversion operation is differentiable.

In addition, we have modified the loss function. When training the original model on our data, we noticed some artifacts that we identified as 'checkerboard artifacts'. These are known to appear when using transposed convolutions (Odena, Dumoulin, and Olah 2016). We have tried known workarounds such as NN-Resize Convolutions (Aitken et al. 2017) but it did not yield better results. We empirically realized that, in our particular case, removing the phase-related loss terms helped reducing these artifacts. Therefore, we removed from (Arık, Jun, and Diamos 2019) the instantaneous frequency loss and the weighted phase loss terms while keeping the Spectral Convergence (SC) term:

$$\text{SC}(s, \hat{s}) = \frac{\| |\text{MEL}(s)| - |\text{MEL}(\hat{s})| \|_F}{\| |\text{MEL}(s)| \|_F}, \quad (11)$$

where $\| \cdot \|_F$ is the Frobenius norm over time and frequency, and the Log-scale MEL-magnitude loss (SC_{log}):

$$\text{SC}_{log}(s, \hat{s}) = \frac{\| \log(|\text{MEL}(s)| + \epsilon) - \log(|\text{MEL}(\hat{s})| + \epsilon) \|_1}{\log(|\text{MEL}(s)| + \epsilon) \|_1}, \quad (12)$$

where $\| \cdot \|_1$ is the L^1 norm and ϵ is a small number.

Finally, our global loss term is:

$$L = \alpha \text{SC}(s, \hat{s}) + \beta \text{SC}_{log}(s, \hat{s}) + \gamma L_{mask}(M, \hat{M}), \quad (13)$$

where α, β and γ are constants used for weighting loss terms. In our experiments, we set $(\alpha, \beta, \gamma) = (3, 10, 1)$, which works well in practice.

Experiments

Dataset

We built a dataset of drums samples coming from various sample packs that we have bought (Vengeance sample packs and others). Overall, we collected more than 40,000 samples across 11 drum categories. All sounds are WAV audio files PCM-coded in 16 bits and sampled at 22050 Hz. Sounds that

were longer than 1 second were removed in order to obtain a homogeneous set of audio samples.

After this preprocessing, the final dataset contains 11 balanced categories (kicks, claps, snares, open and closed hi-hats, tambourines, congas, bongos, shakers, snaps and toms) with 3000 sounds each for a total of 33000 sounds. All sounds in the dataset have a length between 0.1 and 1 second (mean of 0.46 second). In order to validate our models, we perform a class-balanced split between 80% training and 20% validation sets. All the results we present are computed on this validation set to ensure generalization.

As said in previous sections, we compute the Mel-scaled spectrograms of these sounds. To do so, we first pad all waveforms with zeros to ensure a constant size among the whole dataset. Thus, all audio files are 22015 samples long. We also normalize them so that the maximum absolute value of samples is 1. Then, we compute STFTs for all sounds with a Hann window with a length of 1024, a hop size of 256 and an FFT size of 2048. To turn the STFTs into Mel-scaled spectrograms, we multiply the STFTs with the filter-bank matrix we mentioned earlier (Eq. 10).

Experimental setup

Before assembling the two parts of our model to create an end-to-end system, we pre-train each network separately.

We train our CWAE with an ADAM optimizer (Kingma and Ba 2014). The initial learning rate is set to $\eta = 1e^{-3}$ and is annealed whenever the validation loss has not decreased for a fixed number of epochs. The annealing factor is set to 0.5 and we wait for 10 epochs. The WAE is trained for 110k iterations. To obtain a good estimation of the MMD between each q_Z^c and their Gaussian prior, we have to compute enough z . Indeed, it is said in (Reddi et al. 2015) that n in equation 3 should be the same order of magnitude as $d_z = 64$. Therefore, at each iteration, we have to ensure that this criterion is satisfied for each class. We then implemented a balanced sampler, for our data loader to yield balanced batches containing 64 samples for each class. It ensures more stability than a standard random batch sampler. In the end, our final batch size equals $64 \times 11 = 704$.

When training the CWAE, we perform some data processing steps that allow greater stability and performance. First, we compute the log of our spectrograms to reduce the contrast between high and low amplitudes. Then, we compute the per-element means and variances to scale the log-Mel spectrograms so that each element is distributed as a zero-mean unit-variance Gaussian. Indeed, we have noticed that it improves the WAE reconstruction quality.

When training the MCNN, we use the Mel spectrograms without scaling. The initial learning rate is set to $\eta = 1e^{-4}$ and is annealed by a scheduler at a rate of 0.2 with a patience of 50 epochs. The MCNN is trained for around 50k iterations, with a batch size of 128.

Reconstruction

We first evaluate the reconstruction abilities of each part of our system, and the system as a whole. On figure 3, we compare the original spectrogram with both our CWAE's reconstruction and the spectrogram computed on the final output.

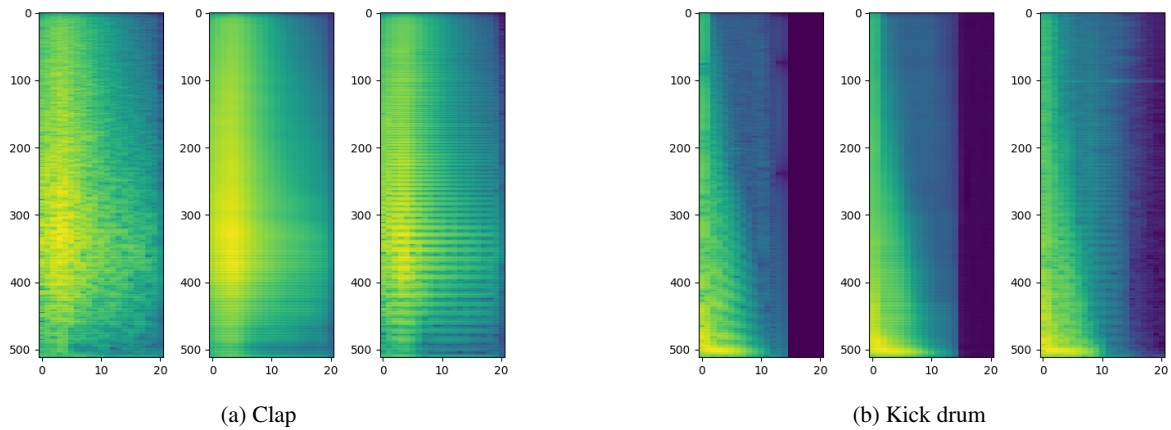


Figure 3: Spectrogram reconstructions of sounds from the evaluation set. From left to right, we have: the original spectrogram, the CWAE reconstruction and the one obtained from the reconstructed waveform (the amplitudes are presented in log-scale for the sake of visibility)

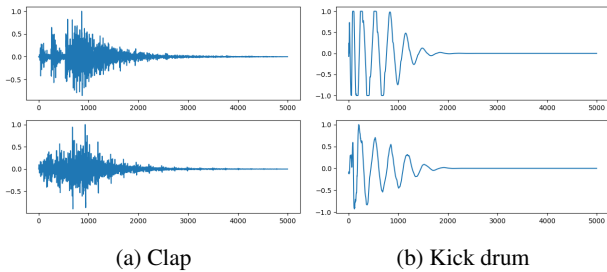


Figure 4: Waveform reconstruction of sounds from the evaluation set. The top row shows the original waveform and the bottom shows the reconstruction after passing the spectrogram throughout the whole system

In both cases, the reconstruction performed by the CWAE is good yet a bit blurry. After passing through the MCNN, we can see some stripes, corresponding to some checkerboard artifact, which periodically affects the waveform. Thus, this appears as a harmonic artifact on the spectrogram. While appearing important on these spectrograms because of the log, the sound is often clean, as shown on the kick reconstruction on figure 4.

More examples are available on the companion website¹, along with audio.

Sampling the latent space

On figure 6, we show generated sounds. We generate them by first sampling a multivariate Gaussian in the latent space. Then, we decode this latent code, conditioned on a given class label and obtain a spectrogram. Finally, this spectrogram is passed to the MCNN which estimates the corresponding waveform. Here, both these sounds are pretty realistic and artifact free. However, sampling the latent space in this fashion does not always yield good sounding

¹<https://anonymous9123.github.io/iccc-ndm>

results. This is because our latent distributions do not really match Gaussian distributions. Also, conditioning on a category does not ensure to generate sounds from this category only. Indeed, some regions of the space will sound close to a hi-hat, even if the class label for claps, is provided to the CWAE. While this can be seen as a drawback, we think that this does not lower the interest because it allows synthesizing hybrid sounds. You can hear additional audio examples on the companion website.

Creative Applications

Interface

For our model to be used in a studio production context, we have developed a user interface. This interface is a Max4Live patch which allows a direct integration into Ableton Live. In this section, we describe how it works and show some screen-shots.

To recall, we pass a (latent code, category) couple (z, c) to the decoder of our CWAE to produce a spectrogram \hat{x} . Then the MCNN generates a .wav file from this spectrogram. However, the latent code z is high dimensional (64 dimensions), so choosing a value for each parameter would be a long and complex process. To facilitate interactivity, we decided to use a Principal Components Analysis (PCA) which aim is to find the 3 most influential dimensions, thus reducing the complexity of the fine tuning process while ensuring a good diversity in sounds. From now on, we denote the PCA dimensions P_1, P_2 and P_3 .

To generate sound through the interface, we provide controllers: First, we provide control over the values for z : an XY pad allows to control P_1 and P_2 and the 'Fine' knob provides control over P_3 . Also, a selector allows the user to define the range of both the pad and the knob. Then, a menu allows the user to set a value for c which comes down to selecting the type of sounds one wants to generate. Finally, the user can use the waveform visualizer to crop out remaining artifacts for example.

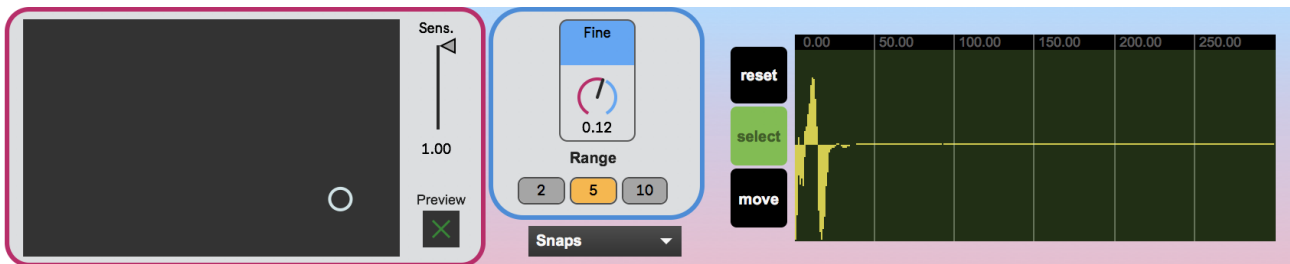


Figure 5: The Neural Drum Machine interface. First, the XY pad on the left controls values for the two most influential dimensions. The "Fine" knob controls the value for the third most influential dimension and can be seen as fine tuning. The range selector controls the range of values available for these three dimensions. Then, a selector allows the user to control which type of sound is generated. Finally, the waveform visualizer on the right allows to trim a sample to play only a particular region.

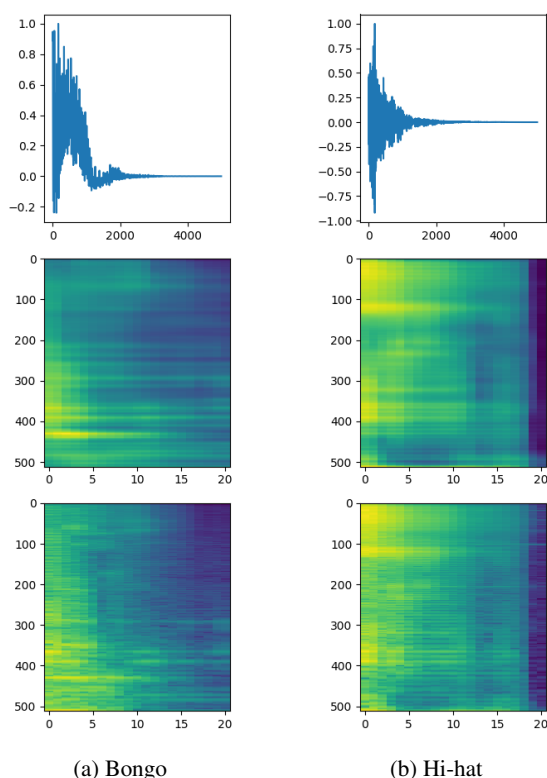


Figure 6: Sounds generated by sampling the latent space. From top to bottom, we have the final waveform, the spectrogram generated by the CWAE and the one corresponding to the waveform (the amplitudes are presented in log-scale for the sake of visibility).

Generation Process

Every time a parameter value changes, a new sound is generated as follows. A python server is listening on a UDP port. This server contains the model and will be in charge of all the computation. When the user modifies the value of a dimension, the Max client sends a message via UDP. This

message contains the values for P_1 , P_2 , P_3 , and the category of the sound. When the server receives the message, it creates the associated latent code z by computing the inverse PCA of (P_1, P_2, P_3) and concatenate it with the conditioning vector. Then the server passes (z, c) to the CWAE decoder which feeds a spectrogram to the MCNN. The obtained waveform is then exported to a WAV file, and its location is returned to the Max plugin. Finally, our plugin loads its buffer with the content of this file and displays it on the visualizer.

Our system can generate sounds with very low latency on CPU (<50ms delay between the change and the sound with a 2,6 GHz Intel Core i7). Once the sound is in the buffer, it can be played without any latency. A demonstration video is available on the companion website.

Impact on creativity and music production

We think that this system is a first approach towards a new way to design and compose drums. Indeed, it is a straightforward and efficient tool for everyone to organize and browse their sample library and design their drum sounds. Despite the parameters being autonomously learnt by the neural network, it is pretty intuitive to navigate in the latent space.

Also, such a tool can be used to humanize programmed drums. It is often claimed that programmed electronic drums lack a human feeling. Indeed, when a real drummer plays, subtle variations give the rhythm a natural groove whereas programmed MIDI drum sequences can sound robotic and repetitive, leaving listeners bored. There are common techniques to humanize MIDI drums such as varying velocities. By allowing the synthesis parameters to vary in a small given range, our system can be used to slightly modify the sound of a drum element throughout a loop. This could, for example, mimic a drummer who hits a snare at slightly different positions.

Conclusion and Future Work

We propose a first end-to-end system that allows intuitive drum sounds synthesis. The latent space learnt on the data provides intuitive controls over the sound. Our system is capable of real-time sound generation on CPU while ensuring a satisfying audio quality. Moreover, the interface we

have developed is studio-ready and allows users to easily integrate it into one of the most used DAWs for electronic music. We identify two axes for improvement: The first one is about the conditioning mechanism that should be more precise and powerful so that each category can clearly be distinguished from the others. The other axis is about developing novel ways to interact with a large latent space to explore its full diversity. Also, similarly to what is achieved on symbolic music (Engel, Hoffman, and Roberts 2017; Hadjeres 2019), we will investigate approaches that let the users specify the controls they want to shape the sounds. This would be an effortless way for novice sound designers to tune their drum sounds and create drum kits on purpose, rather than relying on existing ones. Also, to merge the computation server into the plugin is a required feature for the model to be even more accessible.

References

- [Aitken et al. 2017] Aitken, A.; Ledig, C.; Theis, L.; Caballero, J.; Wang, Z.; and Shi, W. 2017. Checkerboard artifact free sub-pixel convolution: A note on sub-pixel convolution, resize convolution and convolution resize. *arXiv preprint arXiv:1707.02937*.
- [Ark, Jun, and Damos 2019] Ark, S. Ö.; Jun, H.; and Damos, G. 2019. Fast spectrogram inversion using multi-head convolutional neural networks. *IEEE Signal Processing Letters* 26(1):94–98.
- [Berlinet and Thomas-Agnan 2011] Berlinet, A., and Thomas-Agnan, C. 2011. *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media.
- [Bishop and Mitchell 2014] Bishop, C. M., and Mitchell, T. M. 2014. Pattern recognition and machine learning.
- [Chen et al. 2016] Chen, X.; Kingma, D. P.; Salimans, T.; Duan, Y.; Dhariwal, P.; Schulman, J.; Sutskever, I.; and Abbeel, P. 2016. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*.
- [Cover and Thomas 2012] Cover, T. M., and Thomas, J. A. 2012. *Elements of information theory*. John Wiley & Sons.
- [Donahue, McAuley, and Puckette 2018] Donahue, C.; McAuley, J.; and Puckette, M. 2018. Adversarial audio synthesis. *arXiv preprint arXiv:1802.04208*.
- [Engel et al. 2017] Engel, J.; Resnick, C.; Roberts, A.; Dieleman, S.; Eck, D.; Simonyan, K.; and Norouzi, M. 2017. Neural audio synthesis of musical notes with wavenet autoencoders. *arXiv preprint arXiv:1704.01279*.
- [Engel, Hoffman, and Roberts 2017] Engel, J.; Hoffman, M.; and Roberts, A. 2017. Latent constraints: Learning to generate conditionally from unconditional generative models. *CoRR* abs/1711.05772.
- [Esling, Bitton, and others 2018] Esling, P.; Bitton, A.; et al. 2018. Generative timbre spaces with variational audio synthesis. *arXiv preprint arXiv:1805.08501*.
- [Gretton et al. 2012] Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. 2012. A kernel two-sample test. *Journal of Machine Learning Research* 13(Mar):723–773.
- [Griffin and Lim 1984] Griffin, D., and Lim, J. 1984. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 32(2):236–243.
- [Hadjeres 2019] Hadjeres, G. 2019. Variation network: Learning high-level attributes for controlled input manipulation. *arXiv preprint arXiv:1901.03634*.
- [Huang et al. 2018] Huang, S.; Li, Q.; Anil, C.; Bao, X.; Oore, S.; and Grosse, R. B. 2018. Timbretron: A wavenet (cyclegan (cqt (audio))) pipeline for musical timbre transfer. *arXiv preprint arXiv:1811.09620*.
- [Huzaifah 2017] Huzaifah, M. 2017. Comparison of time-frequency representations for environmental sound classification using convolutional neural networks. *arXiv preprint arXiv:1706.07156*.
- [Kingma and Ba 2014] Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- [Kingma and Welling 2013] Kingma, D. P., and Welling, M. 2013. Auto-Encoding Variational Bayes.
- [Odena, Dumoulin, and Olah 2016] Odena, A.; Dumoulin, V.; and Olah, C. 2016. Deconvolution and checkerboard artifacts. *Distill* 1(10):e3.
- [Oord et al. 2016] Oord, A. v. d.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; and Kavukcuoglu, K. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.
- [Peeters 2004] Peeters, G. 2004. A large set of audio features for sound description (similarity and classification) in the cuidado project.
- [Perez et al. 2017] Perez, E.; De Vries, H.; Strub, F.; Dumoulin, V.; and Courville, A. 2017. Learning visual reasoning without strong priors. *arXiv preprint arXiv:1707.03017*.
- [Perez et al. 2018] Perez, E.; Strub, F.; De Vries, H.; Dumoulin, V.; and Courville, A. 2018. Film: Visual reasoning with a general conditioning layer. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- [Prenger, Valle, and Catanzaro 2018] Prenger, R.; Valle, R.; and Catanzaro, B. 2018. Waveglow: A flow-based generative network for speech synthesis. *arXiv preprint arXiv:1811.00002*.
- [Reddi et al. 2015] Reddi, S.; Ramdas, A.; Póczos, B.; Singh, A.; and Wasserman, L. 2015. On the high dimensional power of a linear-time two sample test under mean-shift alternatives. In *Artificial Intelligence and Statistics*, 772–780.
- [Sohn, Lee, and Yan 2015] Sohn, K.; Lee, H.; and Yan, X. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, 3483–3491.
- [Tolstikhin et al. 2017] Tolstikhin, I.; Bousquet, O.; Gelly, S.; and Schoelkopf, B. 2017. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*.

Reshaping Design Search Spaces for Efficient Computational Design Optimization in Architecture

Likai WANG¹, Patrick JANSSEN², Guohua JI¹

1. School of Architecture and Urban Planning, Nanjing University, China

2. School of Design and Environment, National University of Singapore, Singapore
dg1436002@smail.nju.edu.cn, patrick@janseen.name, jgh@nju.edu.cn

Abstract

This paper focuses on the use of using appropriate parametric modelling approaches for computational design optimization in architecture. In many cases, architects do not apply appropriate parametric modelling approaches to describe their design concepts, and as a result, the design search space defined by the parametric model can be problematic. This can further make it difficult for the computational optimization process to produce optimized designs. As a result, the design search space needs to be reshaped in order to allow the computational design optimization process to fully exploit the potential of the design concept on improving the design quality. In this paper, we identify two common types of inappropriate modelling approaches. The first one is related to the design search space that lacks proper constraints, and the second is related to the design search space fixed by the conventional design knowledge. Two case studies are presented to exemplify these two types of inappropriate parametric modelling approaches and demonstrate how these approaches can undermine the utility of computational design optimization.

Introduction

In recent years, computational design optimization has been gaining popularity in architecture because it provides an efficient method for helping architects solve many performance-based building design problems related to material or energy use. By defining a parametric model for the building design and an evaluative model for the building performance, architects or engineers are able to use computational optimization algorithms such as genetic algorithms or direct search algorithms to establish an automated design optimization system. Such systems enable architects to use computers for tedious “number-crunching” where the computer explores the design search space defined by the parametric model. Once the population has been evolved, architects can then identify optimal design variants that can achieve a set of performance requirements. As such, the process of design variants can be driven by building performance. Such design optimization processes can

be referred to as *performance-based* (Oxman, 2009) or *performance-driven design* (Shi & Yang, 2013).

In the research literature, there are numerous successful examples of computational design optimization on improving the building performance. However, in practice, the performance improvements that can be achieved can be limited if only relying on computational design optimization itself.

When applying computational design optimization in architecture, a key task for the architect is to encode their design concept as a parametric model. The parametric model delineates a specific design search space with a family of design variants (candidate solutions) sharing specific characteristics. This model is composed of a set of rules and constraints defining the associative relationships among various parts and components. Ideally, the parametric model should capture the design concept which the architects believe is capable of solving the design challenges.

However, the relationship between the design concept and the parametric model is complex, and for architects, creating such a model in the midst of their design exploration process is often difficult. This is due to that fact that, with different parametric modelling approaches, the particular constraints and rules that the architects define will result in a search space that only includes certain design variants and will also impose biases favour some design variants over others within that search space. As a result, an inappropriate parametric modelling approach may inadvertently end up including too many low-performance designs or excluding the most interesting high-performance design variants from the design search space (Figure 1). In this respect, creating an appropriate parametric model can be more decisive than performing the computational design optimization process itself. This issue is highlighted by Rittel & Webber as follows: “*setting up and constraining the solution space and constructing the measure of performance is ... likely ... more essential than the remaining steps of searching for a solution ...*” (Rittel & Webber, 1973).

Taking this as the point of departure, this paper first identifies two common inappropriate parametric modeling approaches and describes the weakness of the design

search space if the space is defined by these two modelling approaches. In order to overcome these weaknesses, the design search spaces need to be iteratively modified in order to produce an improved design search space for the computational optimization, which can be referred to as *design search space reshaping*. Next, two case studies are presented to crystallise the idea of design search space reshaping and demonstrate how computational design optimization can be undermined by an inappropriate design search space as well as how it can benefit from reshaping.

Design Search Space Reshaping

In the context of performance-based architectural design, exploring design variants is the primary task after the design concept has been defined. With design processes that do not use optimization systems, architects might iteratively generate and evaluate design variants reflecting a particular design concept. The evaluations might use building performance simulations such as computational fluid dynamic (CFD) simulations and energy simulations. By iteratively producing and evaluating new design variants, architects were able to gradually improve design performance (Gero, 1990; Liu, Bligh, & Chakrabarti, 2003). However, such design processes are typically inadequate and inefficient since only a small number of designs could be explored. As a result, the chances of discovering unexpected high-performance designs would be very low.

The emergence of computational design optimization helped to resolve this challenge. With such algorithms, architects were able to define a design search space by encoding their design concepts and then let the computer search the design search space for the optimal design variants. Such automated design optimization methods rapidly become popular in research over the past decade.

While some researchers believe that performance-based design can be fully automated, while others have become aware of the limitation of computational design optimization. Many researchers argue in favour of a human-in-the-loop optimization process, with architect playing a critical role in guiding and filtering the computational design optimization process (Bradner, Iorio, & Davis, 2014; Negendahl, 2015; Stouffs & Rafiq, 2015; Wortmann & Nannicini, 2017). In this respect, Bradner et al. point out that “*the computed optimum was often used as the starting point for design exploration*” (Bradner et al., 2014), and similarly, Wortmann takes computational design optimization as a “*medium of reflection*” (Wortmann, 2018). In reality, however, for those architects who are interested in using computational design optimization either for design exploration or reflection, a challenge they may first encounter is encoding their design concepts with an appropriate parametric modelling approach.

In practice, the problem of inappropriate parametric modelling approaches is not uncommon. Nonetheless, it is often overlooked by architects due to the fact that the parametric models defined by such inappropriate modelling approaches will still result in a computational design optimization process that seems to progressively discover bet-

ter design variants. This will often give architects a false sense of confidence with regards to the actual quality (fitness) of the design variants.

On our observation, there are two key reasons for the misapplication of parametric modelling approaches. The first reason relates to the lack of knowledge on optimization (Wang, Janssen, & Ji, 2018). Many architects who use computational design optimization have little knowledge of the complexity and limitations of such optimization algorithms. The second reason relates to architects’ design fixation, stemming from conventional design knowledge based on past experiences (Wang, Janssen, & Ji, 2019b). Such knowledge can result in the architect overlooking alternative parametric modelling approaches that could result in design variants with significant performance improvements.

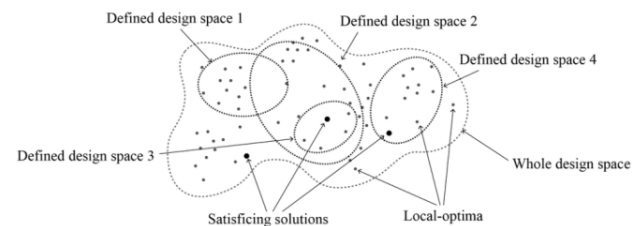


Figure 1. Relationship of design search spaces

In order to overcome these two problems, it requires architects to be more critical on learning from the feedback offered by the computational design optimization process rather than directly using the result obtained by the process. In this regard, the outcomes of the computational design optimization processes can encourage architects to reflect on and improve their parametric modelling approaches in order to reshape the design search space.

In the next two sections, two case studies are presented to illustrate how the above mentioned parametric modelling approaches, as well as the associated problematic design search space, can degrade the result of computational design optimization processes, and how these problematic design search spaces can be modified to allow computational optimization to achieve better results.

Case Study 1

The first case study serves to exemplify the design search space without proper constraints. The design describes a 40-storey high-rise building centred with an atrium. A series of vertical garden voids connecting to the atrium are inserted into the building. The combination of an atrium and vertical gardens are widely used to enhance natural ventilation and moderate temperatures. However, vertical gardens can also occupy a large amount of rental space of the building and increase the overall cost. Therefore, for this case study, the design objective is to search for design variants that can optimize the economic performance taking into account various factors including potential rental profit, façade cost, and construction cost. Thus, high-performance design variants have a rental profit that can significantly outweigh the accumulated cost if the building facades and structures.

In this design, the building was first voxelised in order to insert vertical gardens into the building mass. The floors are divided into multiple fixed-size voxels (Figure 2). Except for the voxels representing the structural cores and the atrium, all other perimeter voxels can be switched between an indoor floor and outdoor void, thereby allowing for complex patterns of interlocking indoor and outdoor space to be created.

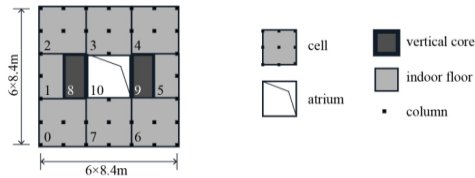


Figure 2. The subdivision of the floor plan into voxels.

Three alternative parametric models were created: without constraints (the first one) and with constraints (the second and the third one). Each parametric model was used to evolve a population of designs with evolutionary algorithms. The models were created in the Rhino-Grasshopper environment, one of the most popular parametric modelling platforms among architects. Design optimization processes were run using Galapagos, an inbuilt optimizer in Grasshopper, which provides a simple genetic algorithm. In order to achieve higher statistical significance, the computational design optimization process was repeated five times for each model.

Design Search Space without Constraints

When encoding this design concept into a parametric model, many designers would prefer a simple and unconstrained parametric modelling approach which independently assigns void-solid conditions for each of the voxels from external parameters. With this approach, the parametric model delineates a design search space with a rich diversity of design variants. At the same time, such parametric models are easy to implement, thereby, making such approaches attractive to architects who often have limited programming skills. However, the downside of using this modelling approach is the extremely large design search space. Moreover, the design search space may also include a great many chaotic design variants which may be too expensive to build. The model is referred to as the Naïve Model.

The drawbacks of this parametric modelling approach and the corresponding design search space can be discovered by running the optimization processes. The first line in Figure 3 presents the result of each of the five design optimization processes. All resultant design solutions have distinct geometric characteristics, and some of these design variants have unexpected geometric features. For instance, the middle one has the merge of voids from consecutive floors allowing for an impressive spatial flowing form. However, from the architectural point of view, most of these can be regarded as infeasible in terms of building geometry.

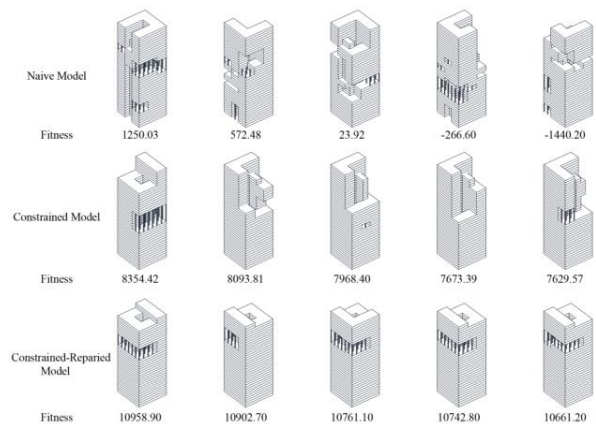


Figure 3. Results of the design optimization processes

In addition to the infeasible building geometry, the design variants can also not meet the required economic performance criteria. Two out of the five resultant designs have a fitness value below zero, which means that these solutions fail to make a rental profit covering the cost, while the other three merely have limited profitability. On the whole, even though these design variants are deemed “optimal”, they fail the basic requirements for feasibility. Meanwhile, even in the sense of supporting reflection, these designs cannot offer much information for architects to extrapolate these to discover some hidden trends or trade-offs of the design problem.

With regards to weaknesses that these design variants have, the unconstrained design search space is the major issue hampering the computational design optimization process to find feasible design solutions. As Rasheed (1998) argued, many parametric models can result in a large proportion of infeasible or invalid design variants in the design search space. This can make it extremely difficult for computational design optimization processes to identify even one single feasible design variant if the design search space is huge. Considering the weaknesses inherited from the unconstrained design search space, the parametric modelling approach needs reformulation by introducing constraints, to reshape the design search space.

Design Search Space with Constraints

Excluding those infeasible design variants within the design search space is necessary to achieve meaningful results from design optimization processes. Hence, direct constraint handling strategies such as *special representations* and *repair functions* (Eiben & Smith, 2004) can be used to achieve such exclusions by avoiding the creation of infeasible design variants. Therefore, the parametric modelling approach was reformulated by applying these constraint handling strategies into the parametric model.

Firstly, with regards to special representations, a set of predefined floor layout patterns were applied to the design (Figure 4), which ensure that the insertion of vertical garden voids only occupies a reasonable size of indoor floors and all vertical garden can connect to the atrium to facilitate natural ventilation. The parametric model applying this

constraint handling design strategy is referred to as the *Constrained Model*. Likewise, based on the same simple genetic algorithm for optimizing the Naïve Model, five design optimization processes were carried out to investigate the effect of the constraints applied in this Model

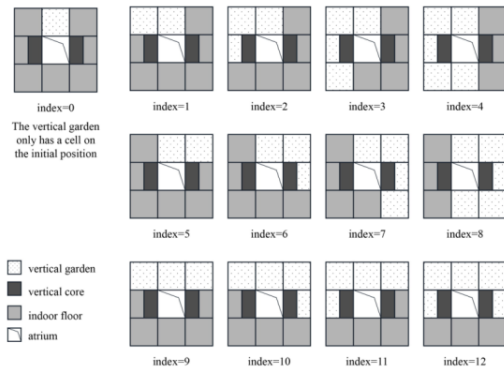


Figure 4. Predefined floor layout patterns

The second line of Figure 3 shows the optimal design variants found across the five design optimization processes. Compared with those obtained with the Naïve Model, these optimal designs have better economic performance in terms of the fitness. In addition, the constraints also make the design optimization processes find design variants with similar geometric characteristics, which help architects extract reliable information from the result.

However, also shown in Figure 3, the building geometries still lack feasibility, which implies that the design search space defined by the Constrained Model still contains a large number of infeasible design variants which prevent feasible designs from being identified. These infeasible design variants have a common undesirable feature: a large hole at the top of the building. It is mostly due to the stacking of floors with the same floor layout pattern. Such floor stacking can result in oversized voids in cases where many voids overlap one another and become merged. In this regard, the constraint embedded in this model is unable to exclude all unwanted infeasible design variants, and the design search space of the Constrained Model needs to be further reshaped (shrunk).

Considering the infeasible feature uncovered by the optimization result, a repair function was applied into Constrained Model, which is able to correct the identified problematic features in the design. The parametric model with the repair function is referred to as the *Constrained-Repaired Model*. The repair function is primarily aimed to control the vertical size of façade voids by preventing voids from stacking.

The repair function is not explicitly enforced under all circumstances, and rather, it is triggered implicitly when violations of the vertical size limit are detected. For this case study, if the void vertically exceeds six stories, the repair function will be activated, and the over-sized void will be iteratively shrunken from the top and the bottom until a suitable height is reached ($a-a'$ in Figure 5). In addition, the repair function also fixes another problematic fea-

ture, where two voids meet at a point on the diagonal resulting in a pair of cross-diagonal voids ($b-b'$ in Figure 5). Likewise, Constrained-Repaired Model was used to optimize the design.

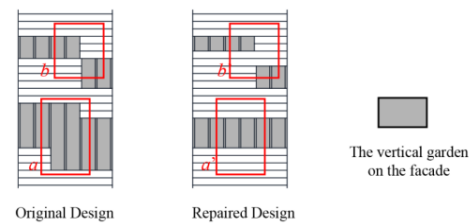


Figure 5. An example of design being fixed by the repair function

The third line of Figure 3 presents the optimal designs found by the design optimization processes with Constrained-Repaired Model. One can find that all these selected design variants have similar geometric features and fitness. On the one hand, the building geometries are more desirable from architectural perspectives compared with those obtained with the Constrained Model. On the other hand, the fitness is also further improved, which implies that these design variants are more profitable. In this respect, the combination of the special representation and the repair function makes the design search space well-constrained. Only with that design search space, can the computational design optimization play a meaningful role in helping architects either understand the design problem or facilitate better decision-making.

With progressively applied constraints into the parametric model, the design search space is iteratively reshaped (shrunk) with the exclusion of a great many infeasible design variants. The first case study not only shows how poorly-constrained design search space can obstacle computational design optimization processes to improve the design but also demonstrates how constraints can be applied to overcome the weaknesses inherited from the unconstrained design search space. However, we should point out the incorporating these constraints will inevitably reduce the variety of design variants in the design search space, which further make the optimization result less unexpected. It is the trade-off the architect should carefully consider.

Case Study 2

The second case study serves to exemplify the parametric modelling approach framed by conventional design knowledge. The design describes a fixed four-story low-rise building centered with a quadrilateral courtyard space. Courtyards have been widely applied in architectural design to improve indoor lighting quality, as it can allow much light to reach the inner part of the building. However, the form and the size of the courtyard volume can result in great differences in its ability to catch sufficient natural lighting for the indoor space. In general, larger courtyards allow for larger natural-lit indoor space, while these also undermine the profitability of the building since the courtyard occupies much indoor space. Thus, when designing

buildings with a courtyard, it is crucial to restrict the size of the courtyard volume while searching for the form of the courtyard can catch as much daylight as possible.

In this case study, three alternative parametric models were created, and these model either stem from paper-based design knowledge or spatial design knowledge. In order to investigate the capability of these models in facilitating computational design optimization to optimize the natural lighting performance of the building, design optimization processes combined lighting simulations were conducted. The simulation was performed by DIVA (Jakubiec & Reinhart, 2011), and Spatial Daylight Autonomy (sDA) was taken as the performance indicator. sDA calculates the percentage of floor area that receives at least 300 Lux for at least 50% of the annual occupied hours (Stern, 2014).

At the same time, in order to restrict the size of the courtyard, the gross floor area of the building was also taken into account. The gross area of the building is normalized to an area ratio by dividing the actual gross area with a target gross area. The area ratio is exponentially decreased along with the decrease in the gross floor area. The area ratio is applied to penalize natural lighting performance. Thus, the value of sDA value is multiplied with the area ratio which is always equal or less than 1. As a result, only when the courtyard with a reasonable size, is the natural lighting performance achieved by the courtyard valid.

The lighting simulation is time-consuming, where each simulation lasts around 1-to-2 minutes. Thus, running the computational design optimization process multiple times in impractical. Hence, for this case study, the optimization process based on each parametric model only ran once. At the same time, an island-model-based evolutionary algorithm (Wang, Janssen, & Ji, 2019a) was applied to optimize the design because this algorithm can yield several diverse design variants, which can facilitate a better understanding of the design search space. In this case study, five parallel search processes were set, so that an equal number of design variants were obtained by the design optimization process.

Design Search Space of Paper-based Design Knowledge

In architecture, the paper-based design method and the rationalist ideology have a great impact on the way how architects describe their design concept, even in digital and computational design environments where higher spatial freedom is allowed by 3D modelling software. One example is 3D parametric modelling approaches that are inadvertently based on 2D conceptual thinking. Such thinking has been deeply rooted in the architectural design knowledge, where building geometry is habitually defined by floor plans and floor height. Therefore, when implementing parametric modelling for courtyard design, the courtyard is often defined by the shape in plan and its height. In this case study, we call the parametric model using this parametric modelling approach the *Plan-based*

Model. For this model, the courtyard shape in plan is defined by the four corner points, and then the shape is extruded to create the courtyard volume.

The first line of Figure 6 presents the five selected design variants found across the optimization process based on the Plan-based Model. In terms of the courtyard shape, these design variants have different shapes but with identifiable shared characters. From all presented design variants, one can find that the courtyards tend to have a triangle or near-triangle shape in plan. This is most likely due to the fact that such triangle shapes can result in a long distance among the courtyard façades so that the mutual shading becomes reduced, thereby, allowing much more natural light to go into the building.

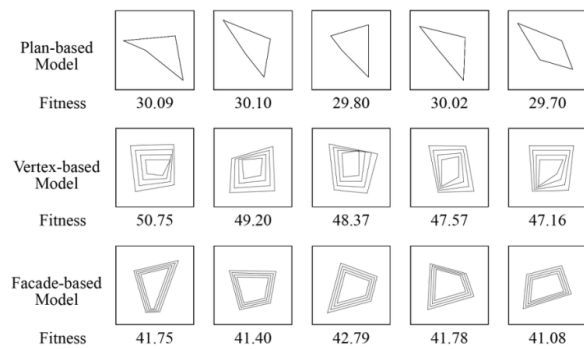


Figure 6. Results of the design optimization processes

However, even the triangle-shape courtyard can reduce the mutual shading, the natural-lit area in the building remain limited. The left column in Figure 7 visualizes the sDA value on each analysis grid based on the building with the highest fitness. Apparently, only the area close to the courtyard can be fully-lit, and the width of the area reduces sharply on lower levels and leave a large area on the lower floors under-lit. It is mostly due to the vertical façades cannot catch daylight when the incident angle of the light is either too high or too low. Hence, in order to achieve bigger improvement by the courtyard, the computational design optimization process should be allowed to search design space outside the one only with the courtyards characterized by vertical façades.

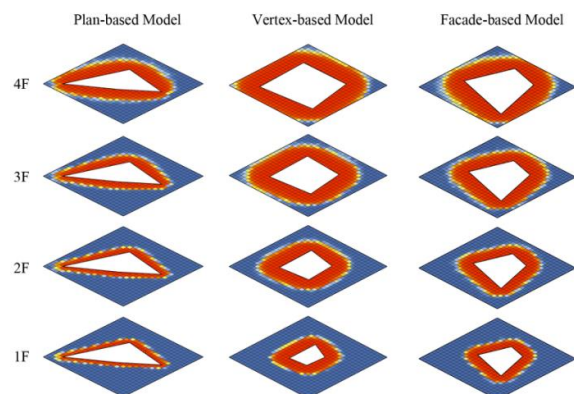


Figure 7. Visualization of sDA analysis

Design Search Space of Spatial Design Knowledge

In order to overcome the weakness stemming from vertical facades, a direct solution is to liberate the inclination angle of the façades as well as the cross-sectional profile of the courtyard. A simple way to free the courtyard from orthogonal volumes is to treat the courtyard volume as a spatial entity. Spatially, the volume of the courtyard is a hexahedron. Thus it can be defined by the eight vertices. Therefore, we reformulated the parametric modelling approach, which defines the courtyard with these vertices, and the corresponding parametric model is called the *Vertex-based Model*. In this model, each vertex is allowed to move independently. By moving these vertices, rich diverse courtyard volumes with inclined and twisted facades can be generated. Such façades are more effective in catching natural daylight from various incident angles and can also reduce mutual shading.

With the Vertex-base Model, the computational design optimization process was performed, which was also based on the same evolutionary algorithm used for Plan-based Model. The second line of Figure 6 shows the optimal design variants found across the design optimization process. The positive effect of using the Vertex-based Model can be immediately noticed in the sharp increase in fitness, where there is approximately 80% improvement on average. It implies that by allowing for higher geometrical freedom, a great many promising design variants are included in the design search space. Even if the inclusion of these design variants enlarges the design search space significantly and thereby may theoretically make the design search space difficult to search, however, the computational design optimization process can quickly identify several design variants with excellent lighting and economic (in terms of the gross floor area) performance.

The middle column of Figure 7 illustrates how the inclined and twisted facades actually improve the natural lighting performance of the building. Compared with the design obtained with the Plan-based Model, the natural-lit area of design obtained with Vertex-based Model is dramatically enlarged. Not only does the top floor can be nearly fully-lit, but also does the bottom floor have a much widened natural-lit area. At the same time, such a significant improvement is achieved without compromising on additional floor area losses.

However, even the natural lighting performance is substantially improved, the use of the Vertex-based Model is not without problems. A major downside comes along with the liberation of the courtyard volume is that most facades are twisted, and such twisted facades are difficult and expensive to construct. As a result, stakeholders could be not interested in such designs with twisted facades, and, if the economic feasibility is the priority for the project, the design variants created by the Vertex-based Model might be problematic. Therefore, economic feasibility and constructability should be considered in order to make the evolutionary result more practical.

With regards to economic feasibility and constructability, we came to the third parametric modelling approach, which can largely satisfy the economic requirement while allowing a significant improvement on natural lighting performance to be obtained. The third parametric modelling approach defines the courtyard volume by its façades, which we call the *Façade-based Model*. Compared with the Vertex-based Model, this model does not allow the façade surface to be twisted, and all façades are a planar surface. Thus, the courtyard volume is changed by rotating the façade vertically and horizontally and moving it along x/y axis. With the Façade-based Model, the façades can still allow being inclined while no twisted façade surfaces will be created.

The third line of Figure 6 shows the optimal design variants found across the computational design optimization process. The fitness indicates that the use of Façade-based Model results in a marked decrease in natural lighting performance compared with those found with the Vertex-based Model due to the exclusion of design variants with twisted facades. However, the planar façade surface makes these design variants more practical in real-world projects. Despite the decrease in performance, the optimal design variants obtained with Façade-based Model can still significantly outperform those obtained with the Plan-based Model due to the inclined façades.

The right column of Figure 7 illustrates the actual natural-lit area of the design variant with the highest fitness value among those found by the optimization process. One can find that the depth of the natural-lit area of this design variant is smaller than that of the design variant found with the Vertex Model. It is mostly because the planar surface is inflexible in controlling the trade-off between catching natural daylight and minimizing the loss of indoor space. In contrast, the twisted façade surface can incline in different angles on each side of the façade. Therefore, such façade surfaces can make the disadvantageous façade section that is blocked by other facades steeper to minimize occupied indoor space, while making the section that can receive much natural light more inclined to maximize the advantage.

For the second case study, the reshaping process of the design search space is in the relatively opposite direction compared with that in the first case study, where the design search space is first significantly enlarged and then slightly shrunken. The reshaping operations used in this case study respectively consider the natural lighting performance and the requirement of economic feasibility. Finally, the Façade-based Model finds a desirable balance between improving natural lighting performance and maintaining economic feasibility.

Moreover, using the models like the Vertex-based or the Façade-based Model can drive the computational design optimization process to maximize the potential of a design concept on improving the concerned performance. Only with this limit known by the architects, can they extrapolate the trends or trade-offs revealed by these optimal design variants to conceive better design concepts. In this

regards, better representation of the design concept lay a more reliable foundation for the reflection on design concepts.

Discussion and Conclusion

This research focuses on the use of appropriate parametric modelling approaches for computational design optimization in architecture. Inappropriate parametric modelling approaches can produce poor design search spaces which cannot well represent architects' design concepts. Such search spaces make the design problem unsolvable either for the reason that the size of the design search space is too large or for the reason that the design search space does not include high-performance design variants. As a result, the potential of a design concept on improving the building performance cannot be efficiently explored by computational optimization.

With two presented case studies, this research identifies and exemplifies two sources that can cause the application of inappropriate parametric modelling approaches. Two case studies also show how these approaches can degrade the usefulness of computational design optimization. The first one stems from the ignorance of the search complexity of the design problem characterized by huge unconstrained design search spaces. In the first case study, even the design search space of the Naïve Model fully covers the design search spaces of the Constrained and the Constrained-repaired Models. However, the enormous infeasible design variants conceal the design sub-spaces characterized by the Constrained and the Constrained-repaired Model for the computational design optimization process to explore.

The second one stems from the design fixation inherited from the conventional design knowledge. In architecture, even 3D modelling techniques have been widely spread, the design knowledge characterized by orthogonal geometries persist. It is also pointed out by Menges as follows: “*In the history of architecture and construction ground breaking technologies have often been initially employed to facilitate projects that were conceived through, and indeed embraced, well established design concepts and construction logics. There is ample evidence of this inertia in design thinking in the context of technological progress*” (Menges, 2007). As a result, fixed by such conventional design knowledge, the design search space is also restricted, and as mentioned earlier, such design search space could not include the high-performance design variants.

In order to address the weakness inherited from the problematic design search space, the design search space needs to be reshaped, which requires the architects to be more critical on their parametric modelling approaches. Moreover, computational design optimization is also indispensable in such a reshaping process. Computational design optimization can help exhibit the design search spaces because the weakness in the design search space is often not explicit, and reaching to the limit of the design search space with computational design optimization can make the problems more “visible”.

To conclude, using computational design optimization methods do not guarantee the architectural design can be truly optimized. One of the crucial promises of performing valid computational optimization is applying appropriate parametric modelling approaches, as inappropriate parametric modelling approaches can end up with “garbage-in-garbage-out”. By reflecting the parametric modelling approaches, the iterative reshaping process towards the design search space not only helps the architects to obtain a better result with computational design optimization but also offer them insight into the performance implication of their design concepts, with which building performance can become a catalyst in architects' design synthesis.

Acknowledgments

This paper is supported by the National Natural Science Foundation of China (51378248) and the China Scholarship Council (201706190203).

References

- Bradner, E., Iorio, F., & Davis, M. 2014. Parameters tell the design story: Ideation and abstraction in design optimization. *Simulation Series*, 46(7): 172–197.
- Eiben, A. E., & Smith, J. E. 2004. *Introduction to Evolutionary Computing*. New York (Vol. 2).
- Gero, J. S. 1990. Design prototypes: a knowledge representation schema for design. *AI Magazine*, 11(4): 26–36.
- Jakubiec, J. A., & Reinhart, C. F. 2011. DIVA 2.0: Integrating daylight and thermal simulations using Rhinoceros 3D, Daysim and EnergyPlus. In *Proceedings of building simulation*. Citeseer. Vol. 20: 2202–2209.
- Liu, Y. C., Bligh, T., & Chakrabarti, A. 2003. Towards an “ideal” approach for concept generation. *Design Studies*, 24(4): 341–355.
- Menges, A. 2007. Computational morphogenesis. In *Proceedings of the Third International Conference of the Arab Society for Computer Aided Architectural Design (ASCAD), Alexandria (Egypt)*. 725–744.
- Negendahl, K. 2015. Building performance simulation in the early design stage: An introduction to integrated dynamic models. *Automation in Construction*, 54, 39–53.
- Oxman, R. 2009. Performative design: A performance-based model of digital architectural design. *Environment and Planning B: Planning and Design*, 36(6): 1026–1037.
- Rasheed, K. M. 1998. GADO: A genetic algorithm for continuous design optimization.
- Rittel, H. W. J., & Webber, M. M. 1973. Dilemmas in a General Theory of Planning. *Policy Sciences*, 4(2): 155–169.

- Shi, X., & Yang, W. 2013. Performance-driven architectural design and optimization technique from a perspective of architects. *Automation in Construction*, 32(4): 125–135.
- Sterner, C. 2014. Measuring Daylight: Dynamic Daylighting Metrics & What They Mean for Designers. Retrieved February 23, 2019, from <https://sefaira.com/resources/measuring-daylight-dynamic-daylighting-metrics-what-they-mean-for-designers/>
- Stouffs, R., & Rafiq, Y. 2015. Generative and evolutionary design exploration. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 29(04): 329–331.
- Wang, L., Janssen, P., & Ji, G. 2018. Utility of Evolutionary Design in Architectural Form Finding: An Investigation into Constraint Handling Strategies. In *International Conference on Design Computing and Cognition*. Springer. 177–194.
- Wang, L., Janssen, P., & Ji, G. 2019a. DIVERSITY AND EFFICIENCY: A Hybrid Evolutionary Algorithm Combining the Island Model with a Steady-state Replacement Strategy. In *24th CAADRIA Conference*. 593–602.
- Wang, L., Janssen, P., & Ji, G. 2019b. PROGRESSIVE MODELLING FOR PARAMETRIC DESIGN OPTIMIZATION: An Example of How Parametric Design Optimization Can Support Reflection. In *24th CAADRIA Conference*. 383–392.
- Wortmann, T. 2018. *Efficient, Visual, and Interactive Architectural Design Optimization with Model-based Methods*. Singapore University of Technology and Design.
- Wortmann, T., & Nannicini, G. 2017. Introduction to Architectural Design Optimization BT - City Networks: Collaboration and Planning for Health and Sustainability. In A. Karakitsiou, A. Migdalas, S. T. Rassia, & P. M. Pardalos (Eds.). Cham: Springer International Publishing. 259–278.

The HR3 System for Automated Code Generation in Creative Settings

Simon Colton,^{1,2} Alison Pease,³ Michael Cook¹ and Chunyang Chen¹

¹ SensiLab, Faculty of Information Technology, Monash University, Australia

² Game AI Group, School of Electronic Engineering and Computer Science, Queen Mary University of London, UK

³ Department of Computing, University of Dundee, UK

s.colton@qmul.ac.uk a.pease@dundee.ac.uk mike@gamesbyangelina.org chunyang.chen@monash.edu

Abstract

We describe the HR3 system for automated code generation, and its use in creative tasks. We outline the motivations and overall ideology behind its construction, most notably by identifying some distinctions in AI methodology which can be ignored when AI tasks are viewed as code generation problems to be solved. We further describe the nature of the approach in terms of: a programmatic interface to a Java API; production rule-based batch processing of data; on-demand code generation and inspection, and the usage of randomised and meta-level codebases. To support the claim that the approach is general purpose, we describe five applications in three areas normally covered by separate Computational Creativity systems, namely mathematical discovery, datamining and generative art. We end by discussing future directions for the HR3 system and how this project might address some higher-level issues in Computational Creativity.

Introduction

In (Colton, Powley, and Cook 2018), we proposed investigating and automating the creative act of software engineering as a major driving force for Computational Creativity research. We further suggested two main principles for undertaking projects where automated code generation was a central technique, namely: (i) that this could/should be done to *problematise the world*, i.e., used to introduce new problems/hypotheses/conjectures and creative affordances instead of/in addition to merely solving given problems, and (ii) generated programs should be celebrated as creations in their own right, not just as a means to an end. In this way, automatic code generation could provide a suitable testing ground for cutting edge Computational Creativity techniques such as framing (Charnley, Pease, and Colton 2012) and dialogue generation to convince users of the value of the generated code artefacts, and address difficult philosophical issues in the field, such as (a lack of) autonomy and intentionality in hand-programmed creative systems.

Existing techniques for code generation include automated program synthesis (Gulwani, Polozov, and Singh 2017), genetic programming (Krawiec 2016) and machine learning techniques such as inductive logic programming (Muggleton 1991). These are surveyed in (Colton, Powley, and Cook 2018), and a tentative position is presented that their limitations preclude them becoming the basis for a

general-purpose automated code generation approach. Another contribution to automated code generation is the HR series of theory formation systems. HR1 (Colton 2002) was a mathematical concept formation program employed in discovery tasks such as the invention of integer sequences (Colton, Bundy, and Walsh 2000). HR2 (Colton and Muggleton 2006) was a more general-purpose datamining system, but has mainly been applied to mathematical discovery tasks, often in conjunction with other reasoning systems, for example to classify finite algebras (Sorge et al. 2008).

These systems can be seen as automated code generators, as HR1 and HR2 could produce Prolog code to represent the concepts it invented, and HR2 could also take code as input to generate data instead of reading it from a file. The code could be written in Prolog or Java and could wrap around code from systems like the Maple computer algebra system (Redfern 1999), so HR2 was essentially making discoveries about Maple code (Colton 2004). The HR3 system has been developed from scratch over the last five years to fully embrace automated code generation based on the central approach of the earlier systems. An early report on the design decisions was given in (Colton, Ramezani, and Llano 2014), along with a comparison of HR3 with HR2 and details of two applications. We describe the latest version of the system here, with five new applications, most notably with HR3 being used for the first time in generative art.

Ideology and Motivation

We are developing HR3 as a *general-purpose code generation* intelligent system for tasks requiring creativity. Our long-term aim is for it to undertake any coding task that a human programmer could complete, and coding tasks that people wouldn't be expected to hand-code, e.g., for generating images of imaginary faces. To achieve this, we view as many traditional AI tasks as possible as automated code generation problems. Our ideology also includes attempting to remove the following distinctions, which seem somewhat artificial in a context of automated code generation:

- *Task Distinctions.* Broadly speaking, **generative** AI methods produce valuable artefacts such as paintings, videogames, poems, musical composition, mathematical concepts, etc., which become an object of interaction/consumption/study. In contrast, **analytic** methods produce more information about given artefacts, and **support**

methods provide additional code which makes the whole project work. In a generative art project, for example, we might require software which (a) produces images (generative), (b) provides information about the textures in the images (analytic), and (c) presents certain images based on texture (support). A human programmer could write code for all these tasks, hence we aim for HR3 to do similarly.

- *Domain Distinctions.* Certain AI approaches, like datamining, are domain independent, but the same is not always true of the practical implementations of these approaches. This is particularly the case for the kinds of generative systems seen in Computational Creativity research, e.g., it would be unusual currently to see a music-generating system write a poem or paint a picture. Often systems are further task-localised, e.g., to generate harmonisations of given melodies rather than producing new ones. We aim for HR3 to be domain independent by enabling it to work with data of any type in a domain-independent way.

- *Clarity Distinctions.* There is often a distinction made between ‘black box’ techniques, the results/processing of which are difficult to understand, and more comprehensible techniques. We aim for HR3 to be able to generate both simple programs if necessary for people to understand them (for example, in datamining), and complex programs where some other criteria such as correctness, speed, variety or beauty in the processing and/or output is more important than clarity, e.g., in generative art.

There are other distinctions in AI methodology that we aim to blur, such as the difference between training and testing stages in machine learning applications, which approaches such as online-learning (Fiat and Woeginger 1998) and one-shot learning already address. Ultimately, we aim for another major distinction to be removed, which is that between a program and its programmer. That is, as HR3 can output code, we ultimately aim for it to re-write, augment and enhance parts of its own program, and we briefly discuss potential benefits of this in the conclusions section.

We gain motivation from the meteoric success of deep learning (DL) approaches in AI. These approaches are applied to both analytic and generative tasks and are domain independent. Moreover, as argued in (Colton, Powley, and Cook 2018), DL clearly performs automatic programming, although the output is not code. Deep learning is so powerful because it produces very large (but not overfitting) programs represented as artificial neural networks (ANNs). This comes at the cost of comprehensibility, as it is usually difficult to understand how an ANN performs a prediction, or generates an artefact. Methods to understand ANNs come from visualisation (which led to the huge growth in generative uses of DL), as well as methods akin to human neuroimaging (seeing which parts of an ANN fire for given inputs) and psychology (asking how an ANN views a series of inputs), but these are rarely as specific or comprehensible as those produced by symbolic AI approaches.

Charnley, Pease and Colton (2012) argue that software explaining how and why it made something is important in accepting the software as creative, and Colton, Pease and Saunders (2018) add that communicating authenticity will

likely be required for acceptance of output as valid, in certain areas like poetry. Hence, we believe that more explainable AI systems are preferred in creative settings over black box approaches. As described below, HR3’s operation is not constrained by a rigid representation scheme, and the Java output it produces can, in principle, manipulate data in any way. General code is more flexible than ANNs, hence HR3 could be more task independent than DL, and code is easier to understand than ANN processing, hence HR3 could be as powerful, yet more comprehensible, than DL.

In the next section, we describe how HR3 operates as a Java API in data-centric creative projects. We then illustrate how HR3 operates, by presenting five new applications in three distinct areas, namely mathematical discovery, datamining and generative art. To conclude, we return to the ideology above to see where HR3 adheres, and where improvements are needed. We end by discussing how this project addresses some Computational Creativity issues, and by describing some directions for future work.

Automated Code Generation

HR3 is a Java Application Programming Interface (API) which can be called upon in various ways for creative projects, to automatically build and employ a **codebase** comprising a set of **methods**. Projects with HR3 are data-centric, with each method comprising **procedures** that manipulate a **database** which is either read from a file or generated initially by user-supplied **background methods**. The simplest way to employ HR3 is to write a single Java file adhering to a few constraints. Normally, this file grows and is constantly tweaked during the project, so we think of it as a **sketchpad**. Sketchpads employ the HR3 API in a **codebase generation phase**, followed by a **codebase interrogation phase**. A GUI is available as an Integrated Development Environment (IDE). While sketchpads can be developed in other IDEs like Eclipse, the HR3 IDE allows the staggering of the generation and interrogation phases, so codebases stay in memory while the user alters and executes the interrogation code repeatedly. As codebase generation can be slow, while interrogation isn’t usually, this saves time.

Importantly (and somewhat ironically for a code generation system), no compilable code is generated until requested by the user, which is usually during the interrogation phase. To explain this, we note that in the worst-case scenario, automated code generation – for instance via a genetic programming (GP) approach – must: (i) generate a representation for a new program, e.g., by crossover and mutation of programs represented as trees (ii) translate the program into compilable code (iii) write this to a file (iv) compile the file into a program (v) run the program (vi) collate the output and (vii) analyse the output, e.g., to estimate fitness. The generation of millions of programs in this way can be slow, which is why in GP, there are many optimisations available. With HR3, for efficiency, we avoid stages (i) to (iv). That is, data is manipulated internally in such a way that it contains the output from methods that HR3 has invented. Standalone Java programs are not created, compiled and executed during codebase generation, but are rather produced on-demand during the interrogation phase later.

Production Rule Batch Applications

HR3 starts by executing the user-given background methods expressed as Java code in a sketchpad. The first background method always produces a set of string constants that act as labels for what we call data **records**. The other background methods flesh out the records by each producing an ordered list of tuples (one list per record) by generating data programmatically and/or reading it from a file. For example, in the datamining applications below, the background methods read data from a CSV file. The first background method generates record IDs using the line numbers in the file and the other methods each extract one value per line, producing singleton tuples in the ordered lists. In contrast, in the numerical discovery application below, the first background method generates a set of integers, and the others calculate the divisors and digits of each integer, producing different length tuples for different integers. The two generative art applications below similarly start with an empty database and background methods which generate appropriate data.

The background methods are taken as the **seed** codebase that HR3 will construct all future methods from. The user directs the usage of **production rules** (PRs, described below) which manipulate information about existing methods into that for a set of newly invented procedures. The application of a PR to an existing method generates new output as an ordered list of tuples for each record, and also a new procedure, represented as a tree capturing the series of PR steps used to construct it – see figure 1 for an example procedure. Because quite different procedures can produce the same output, we define a **method** as a pair which contains (i) the data output by the manipulations of the PRs for the method on the database, and (ii) a set of different procedures which produced that output.

Starting with the seed codebase, the repeated application of PRs to existing methods builds up the codebase. **Unary** PRs are applied to one existing method, and **binary** PRs are applied to two. Under normal operation, each production rule is applied to the **batch** of methods (unary) or pairs of methods (binary) in the codebase that the PR hasn't previously been applied to. For a background or generated method, m , and an ordered list of records R , we write $m(r)$ for the output of m when applied to a $r \in R$ and we write $m(R)$ for the ordered list of outputs of m when applied to every $r \in R$, in order. With this notation, the core components of a codebase containing H methods are:

- A set $R = \{r_1, \dots, r_n\}$ of record IDs
- Sets C_i, C_f, C_s of integer, float and string constants resp.
- A set of methods $M = \{m_1, \dots, m_H\}$, where $\forall i$:

$$m_i(R) = \langle \{m_i(r_1), \dots, m_i(r_n)\}, \{proc_1, \dots, proc_k\} \rangle$$

where $m_i(R)$ is an ordered list ranging over $r \in R$, such that $m_i(r)$ is a set of typed tuples of the same length, i.e., $m_i(r) = \{t_0, \dots, t_n\}$ and $\forall t \in m_i(r), \exists l$ s.t. $t_i = \langle c_1, \dots, c_l \rangle$, with each c_i being a member of C_i, C_f or C_s

- A set, E , of procedures for methods, m , where $\forall r \in R, m(r) = \{ \}$

Informally, for every method HR3 invents, it records (a) the output of that method when applied to the database, as an

ordered list of sets of tuples, one set per record, and (b) the set of procedures that generate methods producing exactly this output. HR3 also separately keeps a set of procedures that led to empty outputs for every record in the database.

The main innovation in the HR projects has been the use of production rules which generate data as positive examples of mathematical concepts (in HR1 and HR2) and output by procedures (in HR3). The data generation processes are cumulative, as they manipulate output from existing procedures into the output for multiple new ones. This is more efficient than starting from scratch each time with the data generated from the background methods, and is analogous to how GP approaches cache sub-tree outputs (Keijzer 2004). There are currently 27 PRs, split into six categories, given in the following table with an indication of whether they are unary (1), binary (2) or neither (0).

Logical:	conjunction(2), disjunction(2), existential(1), instantiation(1), inversion(1), negation(2), overlap(2), unifyVariables(1)
Mathematical:	exponential(1), trigonometry(1)
Meta:	cull(0), tag(0)
Numerical:	banding(1), bounds(1), count(1), interArithmetic(2), interNumCompare(2), intraArithmetic(1), intraNumCompare(1), makeFractional(1), round(1)
Programmatic:	interBitwise(2), randomChooser(1)
Statistical:	normalise(1), numSummary(2), rank(1) sampling(1)

A sequence of PR applications specified by the user in the sketchpad are carried out by HR3 during the codebase construction phase. Many PRs have a **parameterisation** specifying different ways they can be applied, e.g., the *numSummary* production rule calculates: min, max, mean, summation, standard deviation and range values. These can be all applied, or a subset specified with a parameterisation. With all possible parameterisations, the PRs produce 49 different types of Java statement. However, this belies a larger number, as there is a different application of the *instantiation* rule per constant in C_i, C_f and C_s , and multiple applications of *unifyVariables*, *overlap* and *existential*, depending on the types in the tuples output by the methods. The PRs in the *Meta* category don't produce new procedures, but rather *cull* selectively reduces the codebase for memory/time-intensive projects, and *tag* labels certain methods, so future PR steps can be applied only to batches of methods tagged appropriately. We include as much processing as possible like this at production rule level, to increase homogeneity.

The application of a unary PR to an existing method m involves first manipulating $m(r)$ for every $r \in R$ to produce a new set of tuples $m'(r)$, and then manipulating the procedure for m into a new one for m' . For each existing method in a batch application of a PR, HR3 cycles through all possible parameterisations of the PR, curtailed by the user if required. As an example, the *existential* production rule is parameterised by an integer which represents a position, p , in the tuples of $m(r)$. For a given record r , it operates by first removing the entry at position p from each tuple

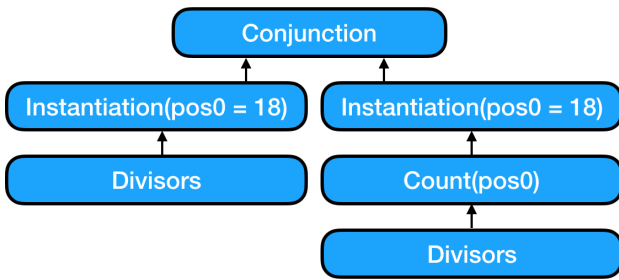


Figure 1: Example procedure for a Boolean method over integers, with PR name and parameterisations in brackets.

in $m(r)$, and then removing any repetitions in the resulting set, recording the remaining altered tuples in $m'(r)$. As another example, the *trigonometry* PR is also parameterised by a position p and cycles through all possibilities for this. It replaces the value x in each tuple at p with $t(x)$ cycling through $t \in \{\sin, \cos, \tan\}$.

Each binary PR manipulates the output of two existing methods, e.g., given methods m_1 and m_2 and record r , the conjunction/disjunction/negation PRs produce tuples thus:

$$\begin{aligned} \text{Conjunction: } m'(r) &= \{t : t \in m_1(r) \text{ and } t \in m_2(r)\} \\ \text{Disjunction: } m'(r) &= \{t : t \in m_1(r) \text{ or } t \in m_2(r)\} \\ \text{Negation: } m'(r) &= \{t : t \in m_1(r) \text{ and } t \notin m_2(r)\} \end{aligned}$$

The storage, retrieval and manipulation of tuples in HR3 has been optimised to make the application of such binary production rules as efficient as possible, as there can be millions of applications of binary rules, with far fewer for unary PRs. With the *negation* PR, HR3 applies it to both pairs (m_1, m_2) , and (m_2, m_1) , and for all three of these binary PRs, HR3 checks that $m_1 \neq m_2$. As another example, the *interArithmetic* PR calculates new tuples by adding/subtracting/dividing/multiplying values in the same position in each pair of tuples of m_1 and m_2 .

As mentioned above, different sequences of PR manipulations of the database can lead to different procedures that produce exactly the same output. It would be redundant to store this output repeatedly, so instead HR3 adds details of any procedure with exactly the same output as that in an existing method m , to the set of procedures that form part of m . It only adds a new method to the codebase if the output, $m(R)$, is distinct from all the other methods currently in the codebase. We have undertaken much experimentation with representation and hashing schemes, as the retrieval of a match to a new output is one of the slowest parts of HR3's process. Whenever HR3 invents a method m for which $\forall r \in R, m(r) = \{\}$, it does not add this method to the codebase, but stores the procedure for it in a set, E , along with others that also produced empty outputs.

On-Demand Code Generation and Inspection

After the codebase construction, HR3 moves onto the user's code in the sketchpad which details how to interrogate the codebase. HR3 can be instructed to turn a particular procedure from a method into executable Java code via the *Spoon API* for Java code generation (Pawlak et al. 2015). In the next section, we show how HR3 forms a codebase of meth-

ods which apply to integers, and from which the user extracts some coincidences. In one run of the sketchpad, HR3 invented a Boolean procedure which checks whether an integer is a multiple of 18 and has 18 divisors, as portrayed in figure 1. The code generated by HR3 for this procedure is given in figure 2. We see that the comments to the method include a definition in a mathematical form which refers to the code for calculating divisors, as given by the user. The comments also give a flattened version of the tree for the procedure and the set of examples (output) that HR3 calculated for it during the codebase construction.

The code in figure 2 is in its most compressed form, and refers directly to the PRs HR3 employed in constructing method number 7255. This presentation is for people who understand HR3's processing. However, the user can specify that HR3 replaces the calls to methods such as `count` and `instantiation` by code which manipulates data directly. If this is not specified, then the methods referred to in the body of `method7255` are included in the Java file, in addition to the background code, which is extracted from the sketchpad, to make the file stand-alone. To make the file executable, a suitable `main` method, which calls `method7255` with appropriate inputs, is also added.

The stand-alone code for a method provides an accurate representation of how it manipulates data, but also allows users to run the code on a larger set of records, e.g., the user could apply `method7255` to the integers between one and a million, even though the codebase was constructed using a much smaller set. The HR3 API enables users to search for methods, e.g., all Boolean methods which output true for a particular record, or all methods with more than 17 positive examples, etc. Users can also employ the API to output code for **conjectures** involving methods of interest, m . For instance, they can ask for **equivalence** conjectures, i.e., all the other methods m' which output the same or similar (based on average per-record set equality) tuples as m over the database, up to a user-given minimum **correctness level**, such as 90%. The Java for equivalences contains the code for particular procedures of m and m' , as well as a `main` method to check equivalence of their outputs, enabling the user to check the conjecture over a larger set of inputs.

The user can also use HR3 to produce **implication** conjectures, where the output from method m' is a subset or superset of that of m , and **mutual-exclusion** conjectures, where methods m' are found which share little or no output with m , again with a minimum correctness level. The set E of empty procedures can also be the source of **non-existence** conjectures, and the user can ask for Java code to check these over a larger set of inputs. The HR3 API includes techniques for presenting, sorting and filtering conjectures, and for checking them against random data, as described below.

Random and Meta-Codebases

HR3 works directly with Java code, rather than a formalism like first order logic, so can in principle produce algorithms for any task, given the correct application of the right production rules. Such an expressive approach means that HR3 generates multiple methods which look different but produce the same output. In some cases, these highlight a

```

// Def: [a] : (18.0=|x1:(divisors(a,x1))|) & (divisors(a,18.0))
// Proc: [Conjunction,[Instantiation,[Count,[divisors,0],0],0,0,18],[Instantiation,[divisors,0],0,0,18]]
// Ex: 180, 252, 288, 396, 450, 468, 612, 684, 828, 882, 972, 1044, 1116, 1332, 1476, 1548, 1692, 1908
public ArrayList<Object[]> method7255_0() {
    ArrayList<Object[]> divisorsAsTuples = divisorsAsTuples(stringData);
    ArrayList<Object[]> instantiation_1 = instantiation(divisorsAsTuples, 0, 0, 18.0);
    ArrayList<Object[]> count_1 = count(divisorsAsTuples, 0);
    ArrayList<Object[]> instantiation_2 = instantiation(count_1, 0, 0, 18.0);
    return conjunction(instantiation_2, instantiation_1);
}

```

Figure 2: Code fragment generated for the procedure given in figure 1.

discovery about the data, but in others the equivalence is due to the nature of the procedures alone. In the latter case, conjectures identifying such patterns are rarely interesting, and it is frustrating to check the conjecture only to find that it expresses an algorithmic tautology. The lack of formalism largely rules out a deductive approach to showing equivalence in these cases. Instead, we implemented techniques to generate a **random codebase** by shuffling the data generated by the original background methods.

When the procedures involved in a conjecture are applied to the random data, if it is still true empirically, it can be fairly safely ignored, as the lack of semantics in the shuffled data indicates a very low probability that the nature of the data is responsible for the pattern expressed by the conjecture. Hence the only alternative is that the procedures themselves force any data into the pattern, and so the conjecture is not interesting. During codebase interrogation, the user can instruct HR3 to employ this method to filter out such uninteresting conjectures. The usage of random codebases is covered in more detail in (Colton, Ramezani, and Llano 2014), and it suffices here to note that the approach can be used on any conjecture type. Moreover, HR3 employs random codebases, along with random sequences of PR applications, to check that the on-demand code it produces for method m outputs the same as the $m(R)$ calculated during the codebase construction phase. Used during the development of new PRs, this has occasionally highlighted mismatches which have been corrected.

Given a **ground codebase**, G , HR3 constructs a **meta-codebase**, G^M , by first taking all the methods of G as the records in G^M , and then giving each distinct tuple of constants in $m(R)$ (for any method m in G) a unique label. It then automatically constructs and adds a single background method, b , to G^M . The output $b(r)$ for a record r in G^M is a set of singleton tuples, with each of these meta-tuples containing a label corresponding to the ground-tuple in G that the ground-method in G (corresponding to r in G^M), outputs. The user can also specify that HR3 adds some additional background methods to G^M that (i) calculate values based on what the ground methods output (ii) express conjectures about the ground methods, and (iii) capture how the methods were constructed, using the procedures in G . For instance, HR3 can add a background method to the meta-codebase which outputs the list of production rule names that went into constructing the ground methods. Once constructed in this fashion, G^M is ready for the codebase construction stage, and HR3 can construct meta-methods which highlight discoveries about the ground codebase.

Example Applications

We aim here to show robustness to different tasks/domains, rather than how successful HR3 is for a particular application or providing operational statistics, etc. We show our ideology of expressing different AI tasks as code generation problems in action, and highlight the practical usage of HR3 sketchpads to achieve goals in creative projects.

Mathematical Discovery

HR1 and HR2 were both effective in number theory, generating new integer sequences and making conjectures (Colton, Bundy, and Walsh 2000). Applying HR3 to such tasks, we employ the following simple sketchpad file:

```

package projects.maths_discovery.integer_sequences;
import java.util.ArrayList;
import ide.Sketchpad;
import production_rules.PR;

public class IntegerSequences extends HR3Sketchpad {

    public ArrayList<String> makeIntegers(int l, int u) {
        // Code generating integers between l and u
    }

    public ArrayList<Integer> divisors(String n) {
        // Code calculating divisors of n
    }

    public ArrayList<Integer> digits(String n) {
        // Code calculating digits of n
    }

    public void generateCodeBase() {
        generateRecords(IntegerSequences.class,
            "makeIntegers", 1, 1000);
        addBackgroundMethods(IntegerSequences.class,
            "divisors", "digits");

        applyPR(PR.Count);
        applyPR(PR.Conjunction);
        applyPR(PR.Existential);
        applyPR(PR.Instantiation, "useInteger:1,2,3");
        applyPR(PR.Count);
        applyPR(PR.Inversion, "useArity:1");
        applyPR(PR.Conjunction, "repeats:2");
    }

    public void interrogateCodeBase() {
        ArrayList<Integer> methodNums =
            getBooleanMethodsTrueFor("23", "53", "73", "113");
        printSeparator();
        println(methodNums.size() + " methods");
        for (Integer mNum : methodNums) {
            println(mNum + ". " + getDefinition(mNum, 0) + "["
                + getOutput(mNum) + "]");
        }
    }
}

```

This contains three background methods (bodies omitted for brevity), plus a method for generating the codebase and one for interrogating it. The code for generating integers, and calculating the divisors and digits of an integer take the most natural Integer input and output ArrayLists of Integers. The IntegerSequence class uses API methods inherited from HR3Sketchpad, including generateRecords, addBackgroundMethods,

Prod Rule	Progress	Procedures	Different	Boolean	Out-Repeats	Out-Empties	Tuples	Memory (Gb)	Time (ms)	Methods/s
Background	2	2	0	0	0	8771	0.00415	72	28
Count	4	4	0	0	0	9866	0.00441	89	45
Conjunction	10	10	0	0	0	9866	0.00562	95	105
Existential	20	16	6	0	0	10866	0.00876	108	185
Instantiation	50	40	30	5	0	10866	0.00651	131	382
Count	58	47	30	6	0	10903	0.01073	137	423
Inversion	88	77	60	6	0	10903	0.01802	150	587
Conjunction	1994	792	743	646	551	10903	0.08788	324	6154
Conjunction	276917	16897	16817	198056	61959	10903	1.54067	32914	8413

```

1314 methods
15. exists x1 ((x1=x2:(digits(a,x2))) & (x1=x3:(divisors(a,x3))))[1, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, ...]
20. digits(a,3.0)[3, 13, 23, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 43, 53, 63, 73, 83, 93, 103, 113, 123, 130, ...]
22. 2.0=|x1:(divisors(a,x1))|[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, ...]
25. 2.0=|x1:(digits(a,x1))|[10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, ...]
...
53. -(divisors(a,2.0))[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, ...]
54. -(divisors(a,3.0))[1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, 19, 20, 22, 23, 25, 26, 28, 29, 31, 32, 34, 35, ...]
...

```

Figure 3: Example HR3 output from the IntegerSequences.java sketchpad given above.

applyPR and getBooleanMethodsTrueFor, and there are many more available. In the sketchpad, the user specifies generating integers 1 to 1,000 for R , and provides some batch parameters to focus the application of the PRs, i.e., specifying that *instantiation* should only use integers 1, 2 and 3; *inversion* (which finds the complement of tuple sets) should only be applied to methods of arity 1; and the final *conjunction* batch step should be repeated twice. The interrogation of the codeBase in the sketchpad is a puzzle: what do the numbers 23, 53, 73 and 113 have in common?

The output from running this sketchpad is in figure 3. We see that – using a single thread on a 2.6Ghz MacBook Pro laptop – the generation phase took around 33 seconds, and HR3 generated and tested 276,917 procedures (at 8,413 per second), producing 16,897 different methods (in terms of procedure outputs), of which 16,817 were Boolean. 198,056 PR steps generated procedures with the same output as one in a method already in the codebase, and 61,959 steps produced an empty procedure. HR3 answers the puzzle with 1,314 procedures, with six given in figure 3: the integers 23, 53, 73, 113 have the same number of (distinct) digits as divisors, namely 2, making them prime numbers, they all have the digit 3 in them, but are not divisible by 2 or 3.

To take the application further than previously with HR2, we dropped the constraints on the *instantiation* PR, ran the codebase generation again, and altered the sketchpad interrogateCodeBase instructions. In particular, we added code to produce a meta-codebase, G^M , from the ground one G . Recalling that each record in G^M represents a method in G , the background methods in G^M were specified to be both the tuple labels (see above) and the production rule steps for each procedure of m in G . Using the applyPR API call, we applied the *count* and *banding* PRs to build a meta-codebase. Using API calls interrogating G^M and cross-referencing the methods in G , with around 10 lines of bespoke code in the sketchpad, we extracted all methods of G which employed the *instantiation* PR grounding variables to a particular number n which was also equal to the number of tuples output by G . We then interpreted these methods and their output as numerical coincidences.

The output was slim, and we were able to cherry-pick and tweet some of the more interesting coincidences, like: “Did you know that between 1 and 1,000, there are 17 multiples of 17 with the digit 7 in them, and 18 multiples of 18 with

an 8 in them?” and: “Between 1 and 1,000, there are 19 primes with a 1 and a 9 in them, and 54 numbers with a 5 and a 4 in them”. Running the sketchpad repeatedly with different integer ranges, we produced more results, such as: “Between 1 and 10,000, there are 36 multiples of 36 with a 3 and a 6 in them, and 45 multiples of 45 with a 4 and a 5 in them” and: “Between 1 and 2018, there are 18 multiples of 18 with exactly 18 divisors”. It is beyond the scope of this paper (where we are assessing the generality, rather than the power, of the approach) to evaluate this application thoroughly. However, we are currently studying the value (if any) of coincidences for everyday creativity, and aim to use HR3 to find coincidences in text and other media.

Datamining

We re-frame datamining as the automatic generation of triples of algorithms which are related via given data. Standard association rule (AR) mining extracts relationships of the form $a = v_1 \wedge b = v_2 \wedge c = v_3 \rightarrow d = v_4 \wedge e = v_5$. While this representation is useful for understanding discoveries about the data, for it to be used operationally (e.g., to see if it holds for a different dataset), the AR will need to be expressed or interpreted as code. In this context, we see that the left hand side (LHS) of the AR is captured by an algorithm extracting all data records from a database for which the value in column a is v_1 , in column b is v_2 and column c is v_3 , with the algorithm for the RHS similar. The implication of the AR is a third algorithm which relates the LHS and RHS algorithms, in this case checking whether the output of the LHS is a subset of the output for the RHS algorithm.

For various (good) reasons of efficiency, correctness checking and avoiding redundancy, standard datamining is usually limited to finding ARs of the above form, possibly with negation added. HR3 can perform datamining in this standard way by constructing a codebase with only the *instantiation* and *conjunction* PRs on data supplied in a CSV file, then using implication conjecture making at the interrogation stage. However, HR3 is able to construct and investigate a much richer variety of algorithms for the LHS and RHS of ARs, but currently the relating algorithm is fixed to the implication (subset), equivalence (equality) and mutual-exclusivity conjectures mentioned above. For instance, by employing the *negate* PR, HR3 can mine ARs with LHS and RHS such as: $a \neq v_1 \wedge b = v_2 \wedge c \neq v_3$. By also employing

exists, this extends to: $a = b \wedge c = v_3$ and the arithmetic PRs enable constructions such as: $a + b = v_1 \wedge c \neq d$, etc.

Re-framed thus, we applied code generation to datamining two substantial datasets. The first contains the position, size and colour of around 1.9m GUI elements from roughly 10,000 Android app screens. We used HR3 to find ARs linking GUI elements to the score on the Android store. The second dataset contains traces from simple arcade-style videogames, compiled to construct a forward model for the game (Dockhorn and Appeldoorn 2018). We also used HR3 to mine ARs that pay into a forward model for each game. Both applications were successful, and HR3 was able to generate thousands of useful conjectures containing statements of various types as above, interpreted as association rules. The API enabled us to sample the data for efficiency, then run generated Java code over the entire dataset to check the validity of interesting ARs. We used the API to calculate support, confidence and Z-values for each AR, with the latter being very useful in sorting results, as high Z-values often indicate surprising, yet well-supported results. The correctness minimum limit was also very useful for experimentation, and the GUI staggering of codebase generation and interrogation saved much time. In both cases, we used random codebases to discard dull conjectures, which worked very well, removing large numbers of conjectures, without (on inspection) discarding any interesting ones.

Generative Art

To expand the tasks HR3 can be applied to, we looked at pixel-based art of the kind generated in (Sims 1991). Background methods in the sketchpad produce record IDs as (x, y) pixels in ranges specified by the user (normally 250×250), and extract the x and the y coordinates. The code generation phase involves initially applying the *trigonometry*, *exponential*, *interArithmetic* and *conjunction* PRs in batches. This produces thousands of large, incomprehensible, functions (with 50+ nodes in the procedure tree), which calculate an output for each pixel based on its coordinates, using trigonometry, surds, exponentials and arithmetic. Codebase construction ends with (a) the *makeFractional* PR removing integer parts of outputs, (b) *normalise* mapping outputs to integers in the range 0 to 255, and (c) *overlap* constructing methods which output triples of these integers. Tagging is employed so only methods output by the previous PR step are employed in the next one, which accelerates a search for more complex procedures, required here.

Around 20 lines of code were added to the sketchpad to take each method outputting triples, and interpret the output as (r, g, b) values in a `BufferedImage` object. For any images of interest, the user generated Java code for the corresponding method, which produced larger (4000×4000 pixel) images for high-res printing and screen display (see figure 4(a)). The intended artwork for this project (entitled *Style Please* as a pun on the phrase ‘Style Police’) was a montage of a face which changes styles over time. Meta-level codebase generation was employed to collate sets of images in a particular style. In particular, the outputs from the ground methods were processed using the *banding* PR at the meta-level, followed by the *count* PR and another

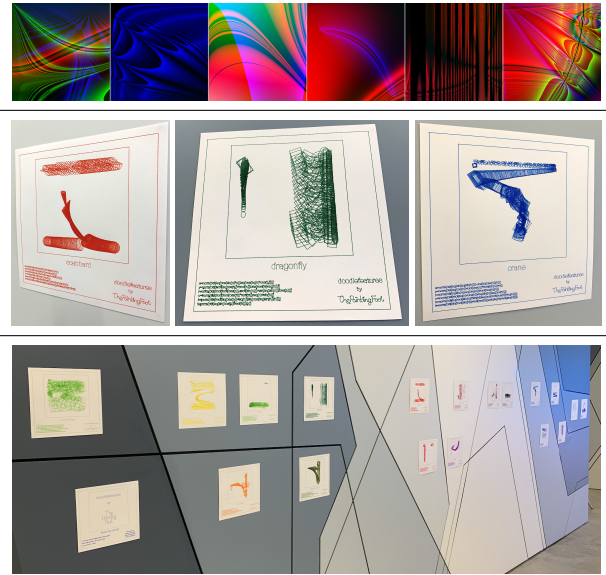


Figure 4: (a) pixel-based generated images (b) the ‘scabard’, ‘dragonfly’ and ‘crane’ plotted artworks from HR3 generated images (c) pop-up exhibition ‘DoodleFeatures’.

banding step. This identified images (i.e., sets of (r, g, b) triples in the method outputs) where, for example, the number of green(ish) pixels was higher than that of red ones, and many other visually obvious styles, such as greyscale, rough/smooth textured, monotone, etc. 100,000 images of (100×100) pixels were produced to provide material for 50 different styles of montage, and the artwork was cycled through them on a 3m by 2m screen for a day.

In another generative art project, we used an AxiDraw plotter to physically produce abstract art pieces, employing HR3 within The Painting Fool project (Colton 2011). Similarly to the pixel-based art, the project sketchpad directed HR3 to produce methods which output a sextuplet for inputs ranging over the integers 1 to n (for a changeable n), rather than coordinates. It used the same codebase generation phase as previously, but with additional **overlap** steps at the end. The sextuplet for an input x were interpreted as the (i) x coordinate (ii) y coordinate (iii) rotation (iv) width (v) height, and (vi) shape type [circle, square, triangle] for a geometric shape that the plotter is able to draw.

Code was added to the sketchpad to (a) render the sequences of n shapes onto a `BufferedImage` to save, and (b) write out the quadruples for each sequence into a Javascript file to be read by the AxiDraw plotter. 100,000 images were passed through a pre-trained ResNet neural model (Krizhevsky, Sutskever, and Hinton 2012) which categorises images into one of around 1,000 classes, corresponding to real-world objects like ‘umbrella’ and environments such as ‘seashore’. All the roughly 100 images which scored 0.8 or above (indicating that ResNet was certain that the images looked like exemplars of the category) were inspected and 18 chosen for a pop-up exhibition called ‘DoodleFeatures’, as portrayed in figure 4. For each, the ResNet category and a representation of the rendering method was added to the Javascript before this directed an AxiDraw plot.

Conclusions and Future Work

In addition to the applications above, HR3 has solved the Countdown Numbers game (Colton 2014), performed invariant discovery in formal methods and addressed dynamic investigation problems (Colton, Ramezani, and Llano 2014). We have concentrated here on breadth of applications, but we plan more in-depth evaluation of HR3's strength for particular applications, and its ability to empower people in a co-creative setting. We believe it significant that code generation has been applied to quite different tasks across application domains. The flexibility of the HR3 approach comes via: casting disparate AI tasks as automated programming; the production rule approach, gaining efficiency by separating code generation from output generation; using meta-level codebases for support tasks, and random codebases to help find the most interesting methods produced.

In the mathematical discovery and generative art applications, additional sketchpad code was needed from the user to complete the project, which indicates room for improvement, as HR3 should be able to generate support code. That said, the meta-codebase generation did help with aspects of the support code, and we have only just begun to explore the affordances of meta codebases. We plan to expand the domains and tasks to which HR3 can be applied, including producing glue code (Liu, Bastani, and Yen 2006); data compression; image filtering; and program synthesis tasks (Gulwani, Polozov, and Singh 2017). This latter application will likely require more goal-based search than is currently implemented in HR3. We also plan to add more automation to the approach, which currently relies too much on the user correctly organising production rule steps in the sketchpad. We aim for a (different) meta-level approach, where HR3 can write its own sketchpads to control code generation, so the user can supply just some background code and/or data.

We also aim for HR3 to be more intelligent in the application of the production rules, for instance, with better abilities to work with functions producing unique outputs, i.e., avoiding PR applications which will certainly lead to empty methods. We also plan for it to output code in different programming languages to Java and for it to improve as a programmer, with (a) more production rules increasing its expressivity, especially with programmatic constructs such as loops and conditionals (b) more sophisticated code styles employing techniques like inlining and variable naming, and (c) more access to relevant data types such as images.

Returning to our ideology, we see that HR3's generated code has been applied across generative, analytic and support tasks, across domains, and (in the generative art example) the image generation code is too large to comprehend, which contrasts with the more comprehensible output in the datamining and mathematical discovery applications. Hence the HR3 implementation blurs the distinctions given above into a continuum. It has problematised the world and introduced artistic affordances by generating stand-alone code inspected in its own right. We plan to add framing abilities so that HR3 can explain and motivate the problems it introduces, and suggest ways to capitalise on new affordances.

Software systems written to be taken seriously as creative in their own right, often suffer criticism that the human pro-

grammer is the creative one, with the software a productivity, or at best, inspiration tool. We plan for future versions of HR3 to alter their own code in an attempt to improve its abilities, and contribute code to other creative AI projects. In this way, we hope to argue that the software is fully independent and hence worthy of being talked about as creative.

Acknowledgements

We wish to thank the anonymous reviewers for their very helpful input. The third author was supported by a Research Fellowship from the Royal Academy of Engineering.

References

- Charnley, J.; Pease, A.; and Colton, S. 2012. On the notion of framing in computational creativity. In *Proc ICCCC*.
- Colton, S., and Muggleton, S. 2006. Mathematical applications of ILP. *Machine Learning* 64.
- Colton, S.; Bundy, A.; and Walsh, T. 2000. Automatic invention of integer sequences. In *Proc. AAAI*.
- Colton, S.; Pease, A.; and Saunders, R. 2018. Issues of authenticity in autonomously creative systems. In *Proc. ICCC*.
- Colton, S.; Powley, E.; and Cook, M. 2018. Investigating and automating the creative act of software engineering. In *Proc ICCC*.
- Colton, S.; Ramezani, R.; and Llano, T. 2014. The HR3 discovery system: Design decisions and implementation details. In *Proc. AISB Symposium on Scientific Discovery*.
- Colton, S. 2002. *Automated Theory Formation in Pure Mathematics*. Springer.
- Colton, S. 2004. Automated conjecture making in number theory using HR, Otter and Maple. *J. Symbolic Computation* 39(5).
- Colton, S. 2011. The Painting Fool: Stories from building an automated painter. In McCormack, J., and d'Inverno, M., eds., *Computers and Creativity*. Springer.
- Colton, S. 2014. Countdown numbers game: Solved, analysed, extended. In *Proc. AISB Symposium on AI and Games*.
- Dockhorn, A., and Appledoorn, D. 2018. Forward model approximation for general video game learning. In *Proc. IEEE Conf. on Computational Intelligence and Games*.
- Fiat, A., and Woeginger, G., eds. 1998. *Online Algorithms: The State of the Art*. Springer.
- Gulwani, S.; Polozov, O.; and Singh, R. 2017. Program synthesis. *Foundations & Trends in Prog. Languages* 4(1).
- Keijzer, M. 2004. Alternatives in subtree caching for genetic programming. In *Proc. of the EuroGP conference*.
- Krawiec, K. 2016. *Behavioral Program Synthesis with Genetic Programming*. Springer.
- Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*.
- Liu, J.; Bastani, F.; and Yen, I. 2006. Glue code synthesis for distributed software programming. In *Advances in Systems, Computing Sciences and Software Engineering*. Springer.
- Muggleton, S. 1991. Inductive Logic Programming. *New Generation Computing* 8(4).
- Pawlak, R.; Monperrus, M.; Petitprez, N.; Noguera, C.; Seinturier, L. 2015. Spoon: A library for implementing analyses & transformations of Java code. *Software: Practice & Experience* 46.
- Redfern, D. 1999. *The Maple Handbook*. Springer.
- Sims, K. 1991. Artificial evolution for computer graphics. *Computer Graphics* 25(4):319–328.
- Sorge, V.; Meier, A.; McCasland, R.; and Colton, S. 2008. Automatic construction and verification of isotopy invariants. *Journal of Automated Reasoning* 40(2-3).

Mechanisms of Artistic Creativity in Deep Learning Neural Networks

Lonce Wyse

Communication and New Media Department
National University of Singapore
Singapore
lonce.wyse@nus.edu.sg

Abstract

The generative capabilities of deep learning neural networks (DNNs) have been attracting increasing attention for both the remarkable artifacts they produce, but also because of the vast conceptual difference between how they are programmed and what they do. DNNs are “black boxes” where high-level behavior is not explicitly programmed, but emerges from the complex interactions of thousands or millions of simple computational elements. Their behavior is often described in anthropomorphic terms that can be misleading, seem magical, or stoke fears of an imminent singularity in which machines become “more” than human.

In this paper, we examine 5 distinct behavioral characteristics associated with creativity, and provide an example of a mechanisms from generative deep learning architectures that give rise to each these characteristics. All 5 emerge from machinery built for purposes other than the creative characteristics they exhibit, mostly classification. These mechanisms of creative generative capabilities thus demonstrate a deep kinship to computational perceptual processes. By understanding how these different behaviors arise, we hope to on one hand take the magic out of anthropomorphic descriptions, but on the other, to build a deeper appreciation of machinic forms of creativity on their own terms that will allow us to nurture their further development.

Introduction

Deep Learning neural networks are notorious for their opacity. We know exactly what is happening at the level of activations and weighted connections between the millions of nodes that may comprise a network, but we are lacking in the analytical tools that would provide human-understandable explanations for decisions or behavior. “Explainable AI” is a prominent goalpost in current machine learning research, and in particular, within the field of Computational Creativity (“CC”) (Bodily and Ventura 2018). Guckelsberger, Salge, and Colton (2017) and others have proposed that the system itself should have a reflective understanding of why it makes creative decisions in order to be considered creative. By understanding the mechanisms that give rise to creative behavior, we might

better be able to build systems that can reflect and communicate about their behavior.

“Mechanisms” have a contested status in the field of computational creativity. For some, there is a reluctance to make the recognition of creativity depend on any particular mechanism, and focus is directed to artifacts over process (Ritchie 2007). This at least avoids the phenomenon plaguing classically programmed AI systems identified by John McCarthy that, “as soon as it works, no one calls it AI any more.” Others (Colton 2008) argue that understanding the process by which an artifact is generated is important for an assessment of creativity.

With neural networks, complex behaviors emerge out of the simple interaction between potentially millions of units. The architectures and unit-level learning algorithms in combination with exposure to data for training allow the networks to configure themselves to exhibit behaviors ranging from classification to media generation. These configurations and behaviors are the “mechanisms” that will be used to explain capabilities that appear akin to human creativity. When mechanisms are emergent rather than explicitly programmed, then their “hidden” nature is not just a methodological choice for framing a definition or establishing criteria for creativity. Understanding how they work requires scientific investigation and is less likely to change our interpretation of their behavior than is unveiling an explicit piece of computer code that generates creative behavior.

The emergent nature itself of neural network mechanisms is relevant to current trains of thought in the CC community. Bodily and Ventura (2018) assert that an autonomous aesthetic, not programmed in by a designer, is fundamental to creative systems. Neural networks also foreground the relationship between perception and generation that has deep roots in psychology (Flowers and Garbin 1989), but has sometimes been neglected in classical AI approaches to programming generative systems. This is because many generative neural networks create media based on the very same mechanisms configured during training on input data from the same domain.

Mechanism is also interesting in complex systems because of the way it can play a role in different kinds of behavior. The mechanism that allows a system to see edges across luminance contrast may also cause illusions in re-

sponse to particular stimuli. Similarly, learning mechanisms that support generalization from scant evidence might enable taking reasonable novel action under unseen conditions in one context, but be recognized as biased decision making when deployed in an inappropriate context.

In this paper, mechanisms in deep learning architectures that give rise to the following 5 characteristics associated with creativity are explored:

- Transformative perception
- Synthesis of different domains
- Sentiment recognition and synthesis
- Analogic and metaphorical reasoning
- Abstraction

These characteristics are a subset of “sub-models of notions used to describe creativity” in Pease and Colton (2011). For the purposes of this paper, no definition of creativity is attempted or necessary, nor are any claims made about whether the mechanisms discussed are “authentically” creative in a human sense. Anthropomorphic terms will be given mechanistic explanations.

Transformative perception

“Make it new” is a phrase Ezra Pound canonized, which refers to a transformative way of seeing that lies at the heart of modern artistic processes. Can neural networks see the world in a unique way based on their own individual experience and share their vision with others?

Deep neural architectures are typically layered, with groups of neurons in one layer connected to the next through “synaptic” weights. A typical example is an image classifier where network inputs are pixel values of images, and output layer cell activations are interpreted as class labels.

In an attempt to understand and characterize what the units and layers are actually computing in terms of the input, a procedure known as activation maximization has been developed (Erhan et al. 2009; Nguyen et al. 2016). The method is very similar to those used to probe human neural response characteristics. Hidden units are monitored, and the input images are manipulated in order to amplify the response patterns of particular individuals or ensembles of units. Systematic exploitation of this idea yielded one of the most prominent image-generating deep learning systems to attract widespread attention beyond the research community, the DeepDream network (Mordvintsev et al. 2015). The manipulated images produced in the service of understanding response characteristics of network units turned out to be striking for their hallucinogenic and dream-like character, and thus an artistic generative technique was born.

Architectures such as the DeepDream network also afford a systematic way to “parametrically” control certain characteristics of the images generated in this way. It turns out (again in a loose analogy to the way the human brain is structured), that peripheral layers tend to respond to “low level” features such as edges, while units in deeper layers

tend to respond to higher-order features such as spatial relationships between lower level feature patterns. As deeper layers are probed, the nodes respond to what we recognize as semantic features. When images are manipulated to maximize the activation level of a particular hidden unit (or ensemble), the results are dependent on the nodes and layers being probed. Images manipulated to maximize activity in peripheral layer nodes yield edge (and other low-level feature) enhancements (Figure 1), while images manipulated to maximize the response of units in deeper layers reveal patterns related to the objects that the networks were trained to categorize at their output.

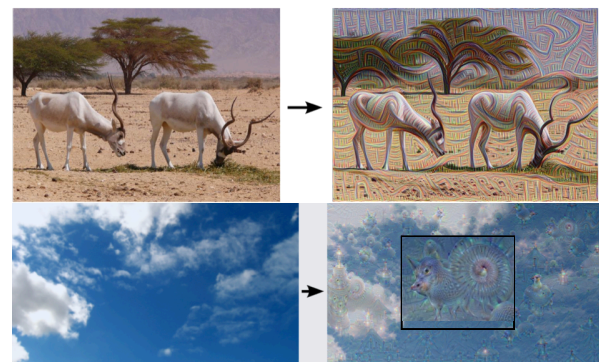


Figure 1. Top: One of the original images presented to the DeepDream network, and the image systematically transformed to enhance highly activated nodes in a peripheral layer. Bottom: An original and a transformed image that enhance highly activated nodes in a deep layer. Inset: A zoomed section of transformed image. (Mordvintsev et al. 2015)

This example illustrates the close relationship between the perceptual and generative capabilities of this kind of deep learning architecture. This is because the novelty in the generated image arises from the technique of exploiting the knowledge in weighted connections learned during perceptual training on input images in order to “read into” the peripheral image during generation. This machinic process of apophenia has been interpreted in other contexts as a dangerous tendency of neural nets to be biased (Steyerl 2018) by interpreting their environments based on their previous limited exposure to data rather than objectively. Apophenia can also be seen as a kind of cross-domain synthesis between a specific image and those previously seen and learned, but a more specific form of cross-domain synthesis will be discussed in the next section.

Cross-domain synthesis

We commonly recognize a distinction between content and style. Content is thought of as “what” is depicted and identified with the subject matter, semantics, or subjects indexically referenced, while “style” is thought of as “how” subjects are rendered through the choice of media and techniques that reflect the process of production or individual perspective. Style is associated with genres, time periods,

and individual artists. A renaissance painter from Florence and an early 20th century Spaniard might both paint a portrait, but each brings a unique perspective to the subject matter through their different styles.

Content and style are independent in the sense that any style can be combined with any content, and thus their combination can be considered one type of “synthesis” across domains. Synthesis is a characteristic that has been considered as a component of creativity across many fields including visual psychology (e.g. Finke 1988). Conceptual Blending is one approach to formal computational models addressing synthesis in the Computational Creativity arena (Pereira and Cardoso 2003).

Gatys, Ecker, and Bethge (2016) showed that the 19-layer VGG-Network (Simonyan and Zisserman 2014) trained to recognize objects learns representations that can be used to separate content from style, as well as for the generation of arbitrary combinations of style and content (Figure 2).

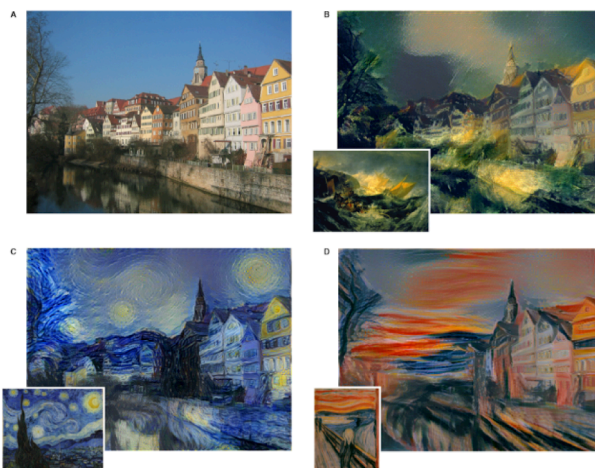


Figure 2. (A) Original photo by Andreas Praefcke rendered in the styles from (B). The Shipwreck of the Minotaur by J.M.W Turner, 1805, (C) The Starry Night by Vincent van Gogh, 1889, and (D) Der Schrei by Edvard Munch, 1893. (Image from Gatys, Ecker, and Bethge (2016) used with permission.)

The mechanism for synthesizing combinations of style from one specific image, and content from another developed by Gatys et al. is reminiscent of the DeepDream network in that during the generative process, an image is presented to the network and then slowly manipulated through the backpropagation of error through the network until it causes activations of the hidden layers to achieve a certain objective. In the case of style transfer however, the objective function for computing error comes in two parts: one is the “content” objective, which is to have the activations at specified layers in response to the formative generated image match the values of those same units in response to the original content image. The second component of the objective is for style. However, rather than try-

ing to achieve a match with actual activation levels of specific hidden layers resulting from the style source image and the generated image, the input is manipulated until there is a match of a statistical measure derived from patterns of activation. The objective based on this second order measure (known as a Gram matrix) maintains correlations between features, but is independent of their spatial location in the 2D representation of the image at the given layers. It just so happens that this measure of statistical correlation between features aligns very closely (although not always completely) with our sense of painterly style as can be seen in the images in Figure 2.

Similar architectures have been shown to work on audio represented as a 2D spectrogram image (Ulyonov and Lebedev 2016), although the results are not as compelling in the audio domain (Shahrin and Wyse 2019). Both Ustyuzhaninov et al. (2016) and Ulyonov and Lebedev (2016) have reported a fascinating aspect of this technique of cross-domain synthesis: that it makes little difference whether or not the network was trained before being used for generation. A shallow network with untrained (randomized) weights can achieve similarly convincing blends of style and content from different images. That is, it turns out the architecture itself, along with the statistical style measure, and the dual style-plus-content objective function, are sufficient for this process of cross-domain synthesis. Note that the link between perceptual (input) process and generation is still present in this architecture because image generation is achieved through the back propagation of error through the same network that responds to image input. Although we might not recognize the process as apophenia without the influence of the learning that we saw in the DeepDream network, the technical process is still one of “reading in” through the mechanism of backpropagation to manipulate an image until it meets criteria defined by an internal representation of input images.

Sentiment

While sentiment, emotion, and affect are not generally considered as defining creativity, they are certainly associated with motivations, processes, and reception of creative artistic works. Understanding emotion is an important part of human social communication, and thus for computational perceptual systems such as face and voice recognition as well. The automatic or parametric control of systems to induce particular emotional responses has been the focus of some generative games (Freeman 2004), and music (Livingston et al. 2007). Picard (1997) considered imbuing computers with the ability to express, recognize, and exhibit emotional behavior (if not to actually “feel” them).

One way to make DNN architectures capable of both categorization and generation of affect would be simply to start with a massive data set with affective labels for supervised training, and hope that there is statistical consistency in the data that the machine can detect. This section discusses a system that developed a kind of “emotional intelligence” without being explicitly trained to do so.

This story starts with a Recurrent Neural Network trained to generate natural language reviews of products. Recurrent neural networks lend themselves to learning and generating sequential data such as speech, music, and text. Data is typically streamed as input to the network one token at a time (e.g. characters from the text of a product review), possibly along with “conditioning” input (such as data from images of products) and the system learns to predict the next character in a sequence (Figure 3).

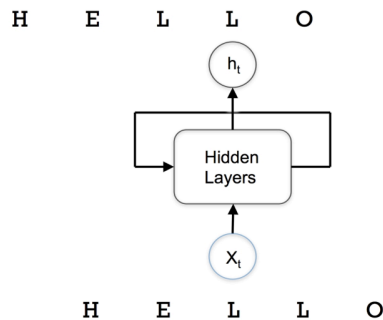


Figure 3. An RNN with recurrent connections training to predict a sequence of characters. The output (top) predicts the next character given the input (bottom), one character per time step.

An RNN can be run in two phases, one as training when the target output is provided by the “teacher” while the network adjusts weights (along each of the arrows in Figure 3), and the other as generation when, after learning, the system produces the next token (in this case a character) in the sequence at the output which is in turn fed back in to the network as the input for prediction at the next step. Karpathy (2015) discusses and interprets creative capabilities of character-generating RNNs.

Radford, Jozefowicz, and Sutskever (2017) were interested in studying what kinds of internal language representations an RNN learns in order to accomplish its predictive task. Specifically, they wondered if the learned representation disentangled high-level concepts such as sentiment and semantic relatedness. They approached the problem as a kind of “transfer learning” paradigm where training on one task (character prediction) is used as a starting point for learning a different task (e.g. sentiment identification).

Radford et al. first trained their predictive system on Amazon product reviews. After training, the RNN can be run in generative mode using each predicted next character output as input at the next time step. It thereby learns to spin text resembling product reviews.

They then consider the trained network as a language model. To test whether the language model internally represents recognizable high-level concepts, a separate linear classifier was trained on the state of the penultimate hidden layer activations during and after processing a product review. Taking sentiment as a particular example, a linear classifier was trained to map this input to the externally provided (supervised) “positive” or “negative” assessment. The performance of the linear classifier is interpreted as

the measure of how well the concept (in this case, sentiment) was captured by the original predictive RNN in the representation of the review.

They discovered that not only was the network able to achieve state-of-the-art performance on rating reviews as positive or negative (compared with networks that had been fully trained specifically for that task), they furthermore noticed that a single node in the representation was responsible for the decision about sentiment. That is, the RNN that was trained only to generate successive characters in a review learned to represent sentiment as a value of a single node in the network. The activation response of that node shows a very clear bimodal distribution depending on the actual sentiment of the review (Figure 4).

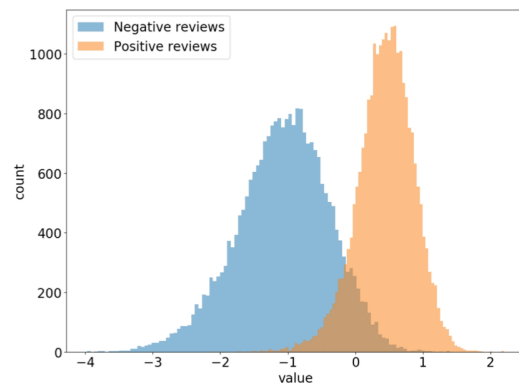


Figure 4. The activation of a single unit in the penultimate layer of a predictive RNN shows a clear binmodal activation pattern corresponding to the sentiment of the review. (Image from Radford, Jozefowicz, and Sutskever (2017) used with permission).

The unsupervised character-level training also provides insight into another important dimension of creative computation in deep learning architectures – the ability of such systems to learn for themselves what aspects of an unstructured environment are significant. That is, it discovered the value of the sentiment representation, the meaning of which is grounded only in its own separate predictive learning task.

Furthermore, the sentiment node can be utilized parametrically in the generative phase of the RNN. By “clamping” it to a value representing the negative or positive assessment and letting the RNN run its character-by-character review synthesis, the reviews it constructs can be clearly recognized as having the desired sentiment characteristic despite all the other variation in a product review.

Analogy and Metaphor

The late 17th century philosopher Giambattista Vico, in his 1725 *The New Science*, recognized the role of metaphor as the foundation of language, reasoning that figurative language precedes literal language and that word meaning only becomes fixed through convention. Some three hundred years later, word2vec models (Mikolov 2013) took a bold step in the field of language modeling to represent

words based only on the context (consisting of other words) in which they appear.

Some sort of vector representation of words is typically used to feed input to neural networks. One way to motivate and understand the word2vec strategy is to start from a “one-hot” baseline representation. A one-hot representation is a vector with a length equal to the number of possible different items to be represented. In this case, that length is equal to the number of words in the vocabulary. Each word is represented with a ‘1’ in its unique position in the vector, and ‘0’s elsewhere. For example, for two particular words we might have:

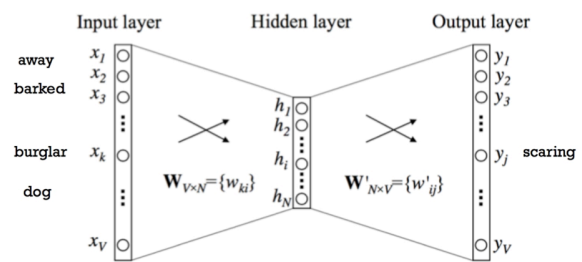
scary: ... 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, ...
 dog: ... 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...

The advantage of one-hot coding is the direct dictionary-like mapping between the vector and the word. However, two issues are immediately apparent. One is the inefficiency of a vocabulary-length vector for each word. The other issue is that the representation does not capture any of the structure that might be inherent in the data such as semantic relatedness or syntactic similarities. To address at least the first problem, a “distributed” code would be much more memory efficient, decreasing the length of each vector, but having more non-zero values in the vector for each word.

A typical way to reduce the dimension of a representation for neural networks is to train an “autoregressive” neural network to learn a reduced representation. An autoregressive neural network simply learns to reproduce its input at the output, but has a hidden layer that is of much lower dimension than the input and output dimensions. By training the network to reproduce all the one-hot representations, the activations in a hidden layer can be interpreted as a new compact distributed representation for the words. The autoregressive network can then function as the encoder (mapping one-hot to distributed representations) to produce the code for use in other neural network tasks and architectures, and as a decoder (mapping the distributed representation back to the one-hot representation to retrieve the human-readable word.

If it were possible to additionally endow the compact distributed codes with some sense of the meanings of the words, then the representation might further assist the networks that will be doing the language processing. The word2vec “Continuous Bag of Words” (CBOW) technique uses the same idea as the autoencoder for reducing dimension, but instead of learning to map just the single word to itself, it learns to map all n-word sequences in a database that surround the target word to the word itself at the output (Figure 5). For example, “The dog barked scaring the burglar away” and “The child enjoyed scaring neighbors on Halloween.” would be two of the (many) thousands of contexts containing “scaring” that would be used to learn to produce the word “scaring” at the output. It is easy to construct the input to train the representations just by adding the one-hot vectors of the context words together. No

sense of ordering is preserved (thus the derivation of the name of the CBOW technique from “bag of words.”)



The dog barked scaring the burglar away.

Figure 5. Training a compact distributed representation (the hidden layer) of the word “scaring” using contexts in which the target word appears.

The encoding and decoding function is preserved by this training strategy. However, now the hidden layer code for a given word not only indexes the individual words, but also embeds information about how the word is used in the language.

This representation is beneficial for training neural networks on a wide variety of natural language tasks. However one of the most impressive demonstrations of the elegance of the representation comes from the original Mikolov et al. (2013). As an m -dimensional vector, the distributed representation of a word is a point in an m -dimensional space where m is much lower than the size of the vocabulary. The question is: does the data have interesting structure in this space, and if so, what kind of structure is it?

It is probably of no surprise that words that have similar semantics occupy nearby points in the representation space based on their similar usage patterns. For example, “frightening” and “scaring” show up in many of the same usage patterns. We can substitute one word for the other in the above example without drastically changing the meaning of the sentences. Because the contexts are similar, the hidden layer learns similar representations.

Simple vector math can be used to explore whether vector operations have any interpretation in terms of the language. For example, taking the vector difference between the points that represent “puppy” and “dog” yields a vector that connects the two points and in some sense “defines” their relationship. What Mikolov demonstrated is that these difference vectors do have semantic meaning. It turns out if we take the same vector that represents the difference between “dog” and “puppy”, and place one endpoint on the point representing “cat”, the other endpoint lands near the point for “kitten” (Figure 6). This ability of vector relationships to capture semantic relationship provides the means for having the system fill in the blanks for A::B as C::? – a kind of analogical reasoning.

The network wasn’t trained for this purpose explicitly, but it self organizes given the task of learning word repre-

sentations based on usage context. Ha and Eck (2017) have done the same kind of vector math on latent vector representations for analogical generation in the domain of drawing images. The Google Magenta group has also done similar work in the domain of music. (Roberts et al. 2018).

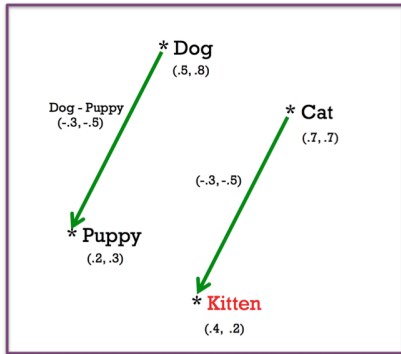


Figure 6. The vector between points representing Dog and Puppy, $(-3, -5)$, when taken from the point representing Cat, points to the neighborhood of Kitten.

Abstraction

Philosopher Cameron Buckner’s (2018) study of deep convolutional networks (DCNN) mechanisms produced a notion of “transformation abstraction.” The layers, convolutions, and pooling mechanisms of DCNNs “jointly implement a form of hierarchical abstraction that reduces the complexity of a problem’s feature space (and avoids overfitting the network’s training samples) by iteratively transforming it into a simplified representational format that preserves and accentuates task-relevant features while controlling for nuisance variation.” That is, classification in DNNs is identified with a process of abstraction.

In the realm of art, there are at least two types of images we call “abstract.” One is comprised of shapes, lines, and colors that are not readily identifiable as any real-world object. Examples include the cube and line “neoplastic” paintings of Mondrian. A second kind of abstract image is comprised of some features that form the basis for an identification as a real-world object, but includes others that are generally not associated with the real world object, and lacks many that are. Duchamp’s *Nude Descending a Staircase No. 2* can be considered of this type.

Tom White’s *Perception Engine* (2018 a, b) foregrounds the notion of abstraction as well as the role of perception in the generative process. One component of the Engine is an image generator constrained to generate combinations of curved lines and blobs. The other is a pre-trained classification network trained on many thousands of example images, each belonging to one of a large collection of labeled categories (e.g. electric fan, golf cart, basketball).

The generator starts with a random combination of the shapes it is constrained to draw, and iteratively manipulates the image to increase the strength of a specific category in a classification network. Figure 7 shows an image generated to maximize the “hammerhead shark” category.

White’s system exposes regions of learned categories in the classifiers where no training examples were provided. The spaces of images that the categorizer classifies together are not otherwise apparent from examining the trained network or the images in the dataset it was trained on.

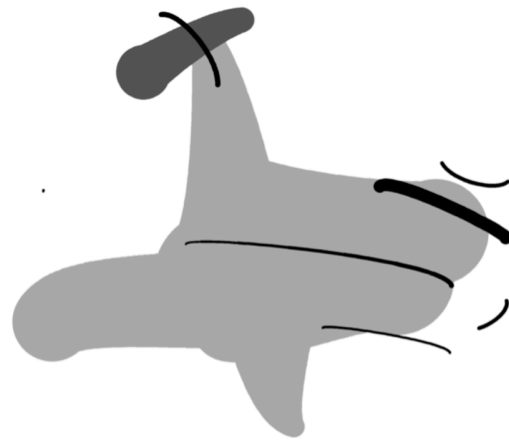


Figure 7. Tom An image from Tom White’s Perception Engine that gets classified as a hammerhead (shark).

The shapes of learned categorical regions can be surprising, and often confront researchers with serious challenges. For example, most trained classifiers, even those that appear to be accurate and generalize well on natural images, are subject to being fooled (Nguyen et al. 2015). “Adversarial examples” (Szegedy et al. 2013), are derived from images that would be classified correctly, but that are modified slightly in a way typically imperceptible to a human, sometimes by only a single pixel (Su et al. 2019), causing misclassification of the modified image.

Adversarial image research focuses on images very near to classification boundaries and reveals the formation of categories very different from those of humans. Tom White’s system on the other hand, explores the weird and wonderful high-dimensional spaces far from boundaries that the system has learned by generalizing during training.

The Perception Engine starts with an unidentifiable “abstract” image (by virtue of its pallet and the randomness), and finishes with an abstract image that contains some local and/or structural features sufficient to generate strong categorical responses in the classifier, while entirely lacking others that real-world objects (including those used to train the networks) have. White says that “the system is expressing its knowledge of global structure independent of surface features or textures” which aligns with the use of the term abstraction. He has also shown that it is sometimes possible for humans to identify the same classification label as do the neural networks, or at least to be able to recognize features in the image that might have caused the neural network classification after its selection is known.

One of the most remarkable features of this system becomes apparent when the resulting generative works are

shown to many different deep-learning classifiers that have been trained on the same real-world data. Despite their different network architectures, the different systems tend to make the same classifications on White's abstractions. That is, if one network classifies the Perception Engine blob-and-line drawing as a "fan", then the image tends to strongly activate the same "fan" category in other classifiers. Generalization patterns in neural networks appear to be general. As a further indication of the nature of the "abstraction" capabilities the networks exhibit, one of his images was generated by optimizing its score for the category of a "tick" insect on 4 neural networks. The image then scores more highly in the tick category than images from the actual human-labeled class of ticks used for validation during training.

It is also interesting to consider the Perception Engine as a kind of "style" machine. The constraints, or 'artistic discipline' of the generator define a style in terms of the types of shapes and lines the system will use to construct images. The target categories can be considered "content," and a series of different drawings by the same generator of different categorical targets would clearly be recognizable as different content sharing a "style," as are most of the works produced by the Perception Engine.

Summary

Taken together, the mechanisms considered in this paper that support the 5 different example behaviors associated with creativity reveal some patterns. The behaviors exhibited by deep learning neural networks are not explicitly programmed, but rather emerge from the simple interaction programmed at the level of nodes and weights. Furthermore, many of the emergent mechanisms identified that serve creative purposes in generative networks arise during training on entirely different and often non-generative tasks. A classification task produced generalization capabilities exploited for abstract image synthesis, a prediction task led to a sentiment representation that afforded parametric control over affect in text generation, and a representational efficiency lent itself to the ability to discover semantic relationships and construct analogies.

Although a wide variety of network types were considered here, all of them use machinery designed for the perceptual processing of media in some way. Some generated output using the exact same substrate used for perception (the DeepDream network) while others used separate, but intimately collaborating perceptual and generative systems (e.g. White's Perception engine). Perceptual capabilities are at the core of deep learning networks, and this paper has illustrated the richness of the connections to creative generation that come from specific computational mechanisms comprising these systems. Perhaps the richness of the connections should come as no surprise given our understanding of human creativity. As Michelangelo wrote "we create by perceiving and that perception itself is an act of imagination and is the stuff of creativity."

Acknowledgements

This research was supported by a Singapore MOE Tier 2 grant, "Learning Generative and Parameterized Interactive Sequence Models with Recurrent Neural Networks," (MOE2018-T2-2-127), and by an NVIDIA Corporation Academic Programs GPU grant.

References

- Bodily, P. M., & Ventura, D. 2018. Explainability: An aesthetic for aesthetics in computational creative systems. In *Proceedings of the Ninth International Conference on Computational Creativity* (pp. 153-160).
- Buckner, C. 2018. Empiricism without magic: transformational abstraction in deep convolutional neural networks. *Synthese*, 195(12), pp.5339-5372.
- Colton, S. 2008. Creativity Versus the Perception of Creativity in Computational Systems. In *AAAI spring symposium: creative intelligent systems* (Vol. 8).
- Mikolov, T., Yih, W.T., Zweig, G. 2013. Linguistic Regularities in Continuous Space Word Representations. *HLT-NAACL*: 746-751.
- Erhan, D., Bengio, Y., Courville, A., and Vincent, P. 2009. Visualizing higher-layer features of a deep network. Dept. IRO, Université de Montréal, Tech. Rep, 4323 2009.
- Freeman, D. 2004. Creating emotion in games: The craft and art of emotioneering™. *Computers in Entertainment (CIE)*, 2(3), 15-15.
- Flowers, J. H., & Garbin, C. P. 1989. Creativity and perception. In *Handbook of Creativity* (pp. 147-162). Springer, Boston, MA.
- Gatys, L. A., Ecker, A. S., and Bethge, M. 2016. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2414-2423).
- Guckelsberger, C., Salge, C. and Colton, S., 2017. Addressing the "Why?" in Computational Creativity: A Non-Anthropocentric, Minimal Model of Intentional Creative Agency. In: *Proc. 8th Int. Conf. Computational Creativity*, 2017. Atlanta, United States.
- Ha, D., and Eck, D. 2017. A neural representation of sketch drawings. *arXiv preprint arXiv:1704.03477*.
- Karpathy, A. 2015. The Unreasonable Effectiveness of Recurrent Neural Networks [Blog post]. Retrieved from <http://karpathy.github.io/2015/05/21/rnn-effectiveness/>
- Livingstone, S. R., Mühlberger, R., Brown, A. R., and Loch, A. 2007. Controlling musical emotionality: An affective computational architecture for influencing musical emotions. *Digital Creativity*, 18(1), 43-53.
- Mikolov, T., Chen, K., Corrado, G., and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

- Mordvintsev, A., Olah, C., Tyka, M. 2015. Inceptionism: Going Deeper into Neural Networks [Blog post]. Retrieved from <https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>.
- Nguyen, A., Dosovitskiy, A., Yosinski, J., Brox, T., and Clune, J. 2016. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In *Advances in Neural Information Processing Systems* (pp. 3387-3395).
- Pease, A., and Colton, S. 2011. On impact and evaluation in computational creativity: A discussion of the Turing test and an alternative proposal. In *Proceedings of the AISB symposium on AI and Philosophy* (Vol. 39).
- Pereira, F. C., and Cardoso, A. 2003. Optimality principles for conceptual blending: A first computational approach. *AISB Journal*, 1(4), 351-369.
- Picard, R. W. 2000. *Affective computing*. MIT press.
- Ritchie, G. 2007. Some empirical criteria for attributing creativity to a computer program. *Minds and Machines* 17:67–99.
- Roberts, A. Engel, J., Raffel, C., Hawthorne, C., & Eck, D. 2018. A hierarchical latent vector model for learning long-term structure in music. *arXiv preprint arXiv:1803.05428*.
- Shahrin, M. H., and Wyse, L. 2019. Applying visual domain style transfer and texture synthesis techniques to audio: insights and challenges. *Neural Computing and Applications*, 1-15.
- Simonyan, K., & Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Steyerl, H. 2018. A Sea of Data: Pattern Recognition and Corporate Animism (Forked Version), in Arrpich, Chun, Cramer, and Sterel (ed.s) *Pattern Discrimination*. University of Minnesota Press
- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.
- Su, J., Vargas, D. V., and Sakurai, K. 2019. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*.
- Ulyanov, D., and Lebedev, V. 2016. Audio texture synthesis and style transfer, [Blog post]. Retrieved from <https://dmitryulyanov.github.io/audio-texture-synthesis-and-style-transfer/>
- Ustyuzhaninov I., Brendel W., Gatys L.A., Bethge M. 2016. Texture synthesis using shallow convolutional networks with random filters. *arXiv preprint arXiv:160600021*
- White, T. 2018a. Perception Engines [Blog post, “Medium”]. Retrieved from medium.com/artists-and-machine-intelligence/perception-engines-8a46bc598d57
- White, T. 2018b. Synthetic Abstractions [Blog post, “Medium”]. Retrieved from https://medium.com/@tom_25234/synthetic-abstractions-8f0e8f69f390

Emolift: Elevator Conversations Based on Emotions

Alejandro Oñate¹ and Gonzalo Méndez^{1,2} and Pablo Gervás^{1,2}

¹Facultad de Informática

²Instituto de Tecnología del Conocimiento
Universidad Complutense de Madrid

{aleonate, gmendez, pgervas}@ucm.es

Abstract

Most of the currently existing narrative generation systems do not consider the task of adding dialogues to enrich the interactions between characters. An important factor to take into account when two characters speak are the emotions they feel and the ones they perceive from the character they are talking to. In this work we present Emolift, an automatic dialogue generation system of short elevator dialogues where emotions are the driving force to build dialogues. We have implemented a system capable of creating dialogues based on the emotions of the characters and the perception of other character's emotions in order to nourish and enrich storytelling systems.

Introduction

Finding something interesting to say to the person standing next to you in the lift is a challenge of everyday life that we all face frequently. Conventional openings like the weather can fill in the awkward silence, but more creative solutions are to be preferred. The present paper explores the assumption that the individual contributions of an agent initiating or participating in such conversations may be generated by assuming a simple underlying purpose generally not related to the actual content under discussion of equalizing the emotional states of the participants in the conversation.

The purpose of this work is to develop an automated dialogue generation system in Spanish where the focus is set on the emotions involved in the conversation. With this objective in mind, we have chosen elevator dialogues, where we usually have conversations without relevant content and with people with little or no affinity. In this way, the variable that most affects the dialogue is the emotional state of the characters. The final goal of this task is to create a component that can be used in storytelling systems to enrich the results produced by traditional generators, which in general tend to neglect this part of the stories. Therefore, for any situation reported in indirect speech, a direct speech equivalent can be generated to provide a better picture about how the characters of the story acted in that situation.

In (Minsky 2010) the author stated that emotions serve as regulators of behavior, since intelligence is characterized, among other aspects, by being able to adjust to a situation. A system should have different outputs for the same entries

depending on the involved emotions. For example, we tend to be more tolerant of the people to whom we have more affection (or affinity), just as our answers generally depend on the way we have received a question.

Therefore, for a conversation to be as realistic as possible, the machine must take the emotional factors into account. Otherwise, if we generate dialogues without considering these parameters, the conversation may be too linear or, on the contrary, give emotional jumps creating totally unusual responses in a real human conversation.

In many storytelling systems, the ability to generate dialogues between characters is still not present (Cavazza and Charles 2005) and practically none of them incorporate emotional factors or characters' personality. Therefore, our most ambitious goal is to create a dialogue generation component for storytelling systems where we can solve this issue: to use the affinity, personality and emotions of the characters with the aim of improving the outputs created by the story generators by adding emotional dialogues between their characters.

In our previous work related to automatic dialogue generation, we approached this problem by using two of the dialogue's modifying parameters: affinity and mood (Oñate, Méndez, and Gervás 2019). Our goal was to be able to generate brief dialogues with a question-answer style to decorate the plan proposals generated by the Charade storytelling system (Méndez, Gervás, and León 2016).

To carry out this task, we take into account the current emotions of the character (base emotions) and the emotions that the character perceives about the speaker (external emotions) with the aim of developing a system capable of creating short conversations (machine-to-machine) like the one that two humans could have in an elevator, considering the emotional values of each character and the impact that the emotions of each character have on the emotional state of the other. This system is called **Emolift**.

In the next sections we present some related work on emotional dialogue generation, followed by a detailed description of the Emolift system accompanied by some examples of its behaviour. We finally present the conclusions of our work, together with the future lines of work to develop a more complete automatic dialogue generator for storytelling systems.

Related Work

Since our goal is to implement a system capable of generating dialogues adapted to the emotions that people feel and the emotions they perceive from their interlocutor, it is necessary to make a brief review of the three phases that our system will face. First, the generation of dialogues, then the analysis of emotions from a text and finally the generation based on emotions.

Dialogue generation

Traum (2017) states that dialogue management systems do not offer a novel interaction. These systems generate dialogues obtained from the analysis of human data, instead of allowing the systems to develop new comprehension and capabilities through the use of dialogues.

In (Oh and Rudnicky 2000) a new approach to generate language for systems of spoken dialogue is presented. For the surface realization task, the authors use an n-gram language model to stochastically generate each expression and observe that this stochastic system works at least as well as the template-based systems. This stochastic generation mechanism has several advantages over template-based ones, such as the response time. Another advantage they point out is that, by using a corpus-based approach, they are directly imitating the language of an expert in real domains, instead of trying to model the sentence generation as a rule, which the authors highlight as a positive factor but that contradicts the conclusions presented in (Traum 2017).

Apparently, this is a very recurring problem, as the Loebner Prize Competition has been used to evaluate the ability of chatbots to fool people making them believe that they are speaking to humans. The chatbots learn to imitate a human using expressions and terminologies that a human would use, while the real objective should be to perform the dialogue generation task for which it has been programmed in the best possible way (Shawar and Atwell 2007).

Emotional analysis

In order to be able to imitate the emotions of humans, it is necessary to analyze how they express themselves in different situations. Therefore, we can see how in the last decade there has been an exponential growth in the field of emotional analysis, and it can be reflected in the number of academic articles published (Lerner et al. 2015).

Thanks to the volumes of information that we expose daily in a public way in social networks, using all kinds of languages (formal, technical, colloquial) and on various subjects, it has been possible to create a wide corpus of emotional information. A clear example is (Pak and Paroubek 2010), a reference when training Neural Networks for the classification of texts according to three types of polarities: positive, neutral or negative.

This type of analysis has quickly spread and led to more specialized studies, since it is no longer enough to know whether a text is positive or negative, but we also need to interpret the emotions that a writer reflects in his/her texts. In this way, researchers began to model mood, subjectivity and emotion detection systems in texts published on social

networks such as (Wang et al. 2012). Even more, some systems use these techniques to combine aspects like the force indicators of opinion, emotion and polarity in order to obtain analysis significantly more accurate than the previous ones (Bollen, Mao, and Pepe 2011).

According to the cues for emotion expression, there are two main methods for sentence emotion recognition: *emotion provoking event* based method and *emotion words* based method. Quan and Ren (2010) use the emotion words based method, which is seen as the most naive approach but also the most popular one. This method consists in extracting the emotional value of a sentence by analyzing each of the words that comprise it. This is the technique we have used throughout this work, which has already been tested for analyzing text in Spanish (Miranda, Luna, and Morillo 2016) and in automatic narrative generation (Delatorre et al. 2017).

Dialogue generation based on emotions

Having analyzed the generation of dialogues and the analysis of emotions in texts, it is time to see where the use of emotions can enrich the generation of texts in general, and the generation of dialogues in particular.

An example where the generation of text based on emotions helped empathize with people is (Mahamood and Renteria 2011). This work uses several Natural Language Generation (NLG) strategies to produce affective medical reports for parents of neonatal babies undergoing medical care. The authors of this work report that all recipients preferred texts generated with affective strategies, regardless of the expected level of stress.

In (Ghosh et al. 2017) a new Affect-LM language model is presented to generate affective conversational texts, which are conditioned by context words, an affective category and a parameter of affective force. This study shows that the model can generate expressive text with varying emotional strength without affecting the grammatical correctness.

With the arrival of personal assistants, research on chatbots and Embodied Conversational Agents (ECA) has intensified in recent years. The most common application areas of the ECAs are education, entertainment, search engines, customer service and support. Polzin and Waibel (2000) present how dialogue behaviors can adjust to the emotional states of users, so it is becoming essential to add interpretation and generation based on emotions to this type of systems.

Finally, and as the greatest success in the generation of emotional dialogues, Zhou et al. (2018) have managed to create a question-answer style system taking into account the five basic emotions that they have considered: Like, Happy, Sad, Disgust and Angry. To carry out this Emotional Chatting Machine, the authors use Fuzzy Logic to discretize the emotional values of the sentences, and offer a response of the emotion obtained. However, this work is only a question-answering system and it is not capable of generating an extensive dialogue.

Emolift System

As we have previously explained, our goal is to elaborate a system capable of generating a dialogue between two characters that meet in an elevator, as a simplified situation that

will later be extrapolated to contextualized dialogues in a story. In this dialogue, we will take into consideration the current state of the emotions felt by each character when they are talking to each other. These emotions are affected by the way our companion addresses us (i.e. the emotions we perceive from their words) (Polzin and Waibel 2000).

It is important to remember that, by taking place in an elevator, these conversations tend to have a peculiar context, and many times they will turn out to be empty conversations, led just by compromise, apart from being really short conversations of approximately 20 seconds long. Because of that, it is not necessary to keep track of emotions, since the conversation takes place in a very short period of time.

This system consists of three fundamental stages. First, the analysis of the emotions transmitted by the sentences, through which we can classify and analyze the sentences of the dialogue. Second, the emotions updater, which will help us modify the base emotions of the characters, combining them with the emotions perceived from every sentence we hear. And, finally, the generation of text based on our updated base emotions, to provide an answer according to our current emotional state. We have addressed these three issues, to have more control and information about what is happening within the conversation. Although the focus of this work is to be found on the last stage of the system, the generation of sentences with emotional value, the three stages are fundamental to be able to generate dialogues based on emotions.

Emotional analysis of a sentence

Since our main goal is the generation of dialogues, and not the analysis of texts itself, and given that high quality data tagged with emotions is hard to obtain as a large-scale corpus, and the annotation of emotions is a very subjective task (Ghosh et al. 2017), we have decided to create a simple system that acts as an analyzer, with the goal of being able to classify the sentences of our dialogue generator.

At the beginning of this work, we made use of two affective dictionaries collecting a total of 3,141 words in Spanish. The first of them (Ferré et al. 2017) consists of 2,266 words, all of them tagged with the same five emotional categories we need. The second dictionary (Hinojosa et al. 2016) includes affective norms for 875 words included in MADS (Madrid Active Database for Spanish) and we chose it because the purpose of this dictionary is to complement the corpus of (Ferré et al. 2017), our main dictionary. As this work was being carried out, a new research was published by the same authors (Stadhagen-González et al. 2018) providing a new version of the affective dictionary that contains three times more words than the other two combined (10,491). By just loading these new words and their values in our system, it was enough to improve its analysis and emotional generation.

The words included in this last dictionary have a value from 1 to 5 for each of the considered emotions according to the intensity with which it transmits that emotion. These emotions are joy, anger, sadness, fear and disgust.

With our dictionary already indexed, the task of analyzing existing emotions in sentences begins. For this, we

have started using the mechanism described in (Eugercios, Gutierrez, and Kaloyanova 2018) for the development of EmoTraductor. We start from the basis that any sentence is devoid of emotion so, if it does not contain any emotional word, the result will be that it is a 100% neutral sentence. Therefore, the emotion of a sentence is based on the combination of the emotions of the words that make up that sentence. So, to analyze the sentence, we must previously analyze each of its words. To carry out this analysis, we will first identify the type of word for each of them.

The relevant words are the names, verbs, adjectives and adverbs. We search these words in our emotional dictionary with the objective of obtaining the emotions that they transmit. Since words have derivations, if the exact word is not found in the dictionary, we compare its lemmatization with the lemmatization of the dictionary words. In the latter case the same search is performed using the stemmer of the word.

Example in our dictionary:

Word: aceptamos (we accept)

Lemmatization: aceptar (infinitive)

Stemmer: acept (remove suffix)

Once we have the emotions for each of the words, we proceed to calculate the emotion of the sentence. Eugercios, Gutierrez, and Kaloyanova propose a calculation based on the arithmetic average of the emotions per word, in such a way that the emotion of the sentence “*Me alegro mucho*” (I am very happy) is calculated by adding the values of the words *alegro* and *mucho* and dividing by three.

This way of calculating emotions is functionally valid, but, from our point of view, it has a drawback. If we perform an arithmetic average, adding extra words with joyful emotion to our sentence will diminish the intensity of that emotion in the sentence. For a human being, naturally, the more you decorate a sentence with happy words or joyful adjectives, the happier it is. For example, if instead of “*Me alegro mucho*” (I’m very happy) we say “*Genial, me alegro mucho*” (Great, I’m very happy) or even “*Genial, me alegro mucho, me hace feliz saberlo*” (Great, I’m very happy, it makes me happy to know it), it is obvious that each of the examples conveys more joy than the previous one. Following the norm of the arithmetic average, less happy words would reduce the final value of the sentence.

Das and Bandyopadhyay (2009) propose a more complex formula, taking into account the value of all the emotions for the calculation of each of them. The value of each emotion is the total value of that emotion (STW_i) multiplied by the words that produce that emotion (N_i) divided by the sum of the values of each emotion (STW_j) and multiplied by their appearances (N_j):

$$SW S_i = \frac{STW_i * N_i}{\sum_{j=1}^5 STW_j * N_j}$$

Evaluating this formula, we can see that the decorating words do not increase the emotion. We propose to find the

Spanish Sentence (<i>English</i>)	Ours	Avg.
Me alegre por ti (<i>I'm happy for you</i>)	3.69	2.12
Me alegre mucho por ti (<i>I'm so happy for you</i>)	3.74	2.01
Me alegre mucho mucho por ti (<i>I'm so very happy for you</i>)	3.78	1.92

Table 1: Comparison of our formula with (Eugercios, Gutierrez, and Kaloyanova 2018) to calculate *joy*

word with the highest value for each of the emotions, and use it as the base emotion for the whole sentence. For each of the remaining words, we take their dominating emotion and use its value to increment the base value of that emotion. We have determined empirically to apply an increase of 10% of its value, always respecting the top limit of 5 points. This way, we get the previous example as follows:

$$S_i = West_i + \sum_{k=1}^n Wres_{ki} * 0.1$$

where S_i is the “Current emotional value of the sentence”, $West_i$ is the “Value of the word with more emotion” and $Wres_{ki}$ is the “Emotional values of the Residuary word in position k ”.

In **Table 1** we can see how our formula gives greater value to decorated sentences, where a human expresses greater joy.

Emotional dialogue generation

Once we have our emotional dictionary, and having defined the procedure of emotional analysis for sentences, we can start generating dialogues based on the emotions of the characters. To carry out this task, we have opted for a system based on templates, which is more controlled and allows us to evaluate and put the emotional analysis into practice.

Our goal is to create a dialogue generation system, not an emotional chatbot, so there is no intervention of a user as an external agent. This point is crucial, since our knowledge base must be more concrete and concise. This allows us to focus on clear objectives and therefore it is not necessary to prevent atypical inputs in the system.

For this reason, we have ruled out conventional template systems such as AIMS¹ or RiveScript², and we have developed our own template system, where the input recognition does not prevail but the emotional variability of the output.

Each sentence of our templates makes use of a nomenclature that allows us to add alternatives when using words or expressions that generate emotions. The system will choose the most appropriate alternative in order to get the sentence that best suits the current emotion of the character.

Following the example of the previous section we could create a template like this:

¹<https://legacy.pandorobots.com/botmaster/en/home>

²<https://www.rivescript.com/>

Original Template

```
<|Genial,|Fantástico,|Estupendo>me alegre <por ti
|mucho ><|, me hace feliz saberlo>
```

Translated Template

```
<|Cool,|Fantastic,|Great>I'm glad <for you |really
><|, it makes me happy to know it>
```

To make the selection of the sentence that has the most emotional similarity with the current emotional state of the character, the system analyzes each section (included in []) and obtains the alternatives delimited by the separator (the character —). Note that the first and last sections begin with this separator, which allows us to indicate that among the available alternatives there is the option of not selecting any expression (select empty string).

Once we have identified each of the sections and their options, we generate all the possible combinations of sentences using the Cartesian product. This allows us to obtain a great variety of responses with different emotions for a simple sentence like this one. The options of this example and the emotional value of each of them would be:

```
Me alegre por ti
joy: 3.69, anger: 1, sadness: 1, fear: 1, disgust: 1
```

```
Me alegre por ti, me hace feliz saberlo
joy: 4.83, anger: 1, sadness: 1, fear: 1, disgust: 1
```

```
Genial, me alegre mucho, me hace feliz saberlo
joy: 4.84, anger: 1, sadness: 1, fear: 1, disgust: 1
```

```
Fantástico, me alegre por ti, me hace feliz saberlo
joy: 4.87, anger: 1, sadness: 1, fear: 1, disgust: 1
```

```
Fantástico, me alegre mucho
joy: 4.51, anger: 1, sadness: 1, fear: 1, disgust: 1
```

```
Estupendo me alegre mucho
joy: 4.32, anger: 1, sadness: 1, fear: 1, disgust: 1
```

Following this process, we have developed a corpus of brief dialogues typical of an elevator conversation. For each of them, we have added variations and alternatives like those shown in the previous example.

From now on, we will use an example of a real dialogue, in which two neighbors are in the elevator and one asks the other how he/she is. Among the possible answers, we have the following template, with which, after applying the Cartesian product, we obtain 1,008 possible combinations, with 878 different emotional combinations. This is because some combinations produce exactly the same emotion, since there are words in several sections that produce neutrality (all values set to 1).

Original Template

<Sin duda |Naturalmente |Normal |Efectivamente
|Buena idea |Buena decisión |Fantástico |Genial
|Maravilloso |No esta mal |En fin |Claro |Que envidia
|En serio? >, <una |cualquiera ><se aburre |no puede
estar >tantas horas <|sin salir |sin salir de casa |en
casa |sin hacer nada |parada ><|que se agobia |que se
cansa >.

Translated Template

<Definitely |Naturally |Normal |Effectively |Good idea
|Good decision |Fantastic |Great |Wonderful |Not bad
|Anyway |Of course |I envy you |Seriously? >, <any-
one |whatever ><gets bored |can not be >so many
hours <|without leaving |without leaving home |at
home |without doing anything |stand ><|because it is
overwhelming |because he gets tired >.

To carry out the elaboration of the dialogue, we begin by identifying the relevant emotions of the current state of the character. Usually, one emotion tends to stand out, but in the case of negative emotions, they tend to stay together. The next step is to analyze the emotional value of each of the combinations of the sentences of our template. From this repertoire of alternatives, we select those in which the same emotions predominate as in the character. Among these, we look for the closest value to the dominating emotion of the character, and then the other emotions following their order.

As stated previously, the quality and precision of our generator depends directly on the quality of both the emotional dictionary and our collection of dialogue templates.

Impact and influence of emotions

The theory of classification states that emotions arise from the interaction between the things we consider important and the events that happen in our environment, which stimulate us (Marsella et al. 2010).

That is why human beings gather events that vary the values of our emotions. It should be noted that a simple conversation in the elevator is considered an interaction and may affect to a greater or lesser degree our emotional state, depending on the content and emotions of the dialogue.

In addition, our emotional state influences the way we talk. Based on the theory of classification, when someone speaks to us, the way they do it and the terms that they use affect our emotions and our response. In turn, the way that person speaks to us is conditioned by their own emotions.

Therefore, we can conclude that, if we speak according to our emotions, and if the way people speak to us affects our response, then the way we speak depends directly on our current emotional state and the emotional state we perceive of the speaker.

Following this assumption, our system must not only take into account the emotions of the character that will speak when generating their response in the dialogue. To be more realistic, we must take into account the response received earlier in the dialogue, with the objective of updating the

character's current emotions to select and respond according to the new value of these emotions.

In order to determine how external emotions influence our own emotions, we need to understand how emotions affect one another. For example, sadness is usually affected by emotions such as anger or fear (Bolles and Fanselow 1980). In general, emotions with greater intensity tend to predominate over the others. The relationship between two or more emotions that can suppress the intensity of other emotions is known as inhibition. Some models use techniques that inhibit emotions that are weaker and contrary to those that have greater intensity (Velsquez 1997). For example, sadness would tend to inhibit joy if it had a greater intensity.

To develop our system, we have used a similar technique: opposite emotions are inhibited according to their intensity, but negative emotions (fear, anger, sadness, disgust) have preference over positive emotions (joy) because they are more persistent and dominating emotions.

Therefore, the first thing we do before updating the emotions of a character is to identify if the dominating emotion of the current state of the character is joy or a negative emotion. Similarly, we extract the dominating emotion from the emotions we perceive from the response. If we are cheerful and have a happy interaction, the value of this emotion will increase more than if we have it while we are sad or furious. Likewise, the rest of the emotions will lose more or less weight when getting the answer to the interaction according to our current state. To illustrate these examples, we show the formulas we have obtained after adjusting the thresholds empirically.

If Joy me and NOT Joy sentence:

$$Rel_{new} = Rel_{me} + Rel_{sentence} \cdot 0.4$$

Relevant emotions of the sentence get a small increment, given that my state is contrary to it

$$Joy_{new} = Joy_{me} \cdot 0.8$$

As a result of being an opposite emotion, the negative ones are reduced by 20% as well

If NOT Joy me and NOT Joy sentence:

$$Rel_{new} = Rel_{me} + Rel_{sentence} \cdot 0.6$$

Relevant emotions of the sentence get a big increment, given that my state is complementary to it

$$Joy_{new} = Joy_{me} \cdot 0.5$$

Everything is not-joy, so joy is worth just one half

If Joy me and Joy sentence:

$$Joy_{new} = Joy_{me} + Joy_{sentence} \cdot 0.6$$

$$Other_{new} = Other_{me} \cdot 0.5$$

The remaining emotions are reduced by one half

If NOT Joy me and Joy sentence:

$$Joy_{new} = Joy_{me} + Joy_{sentence} \cdot 0.4$$

$$Other_{new} = Other_{me} \cdot 0.8$$

Getting a joy sentence, the negative ones are reduced by 20%

The stronger emotions tend to dominate the conversation, directly influencing the emotions of the listener. This causes that, during the conversation, the emotions of both characters tend to balance.

Emotions have a temporary life that depends on the weight and importance of the event. In our case, being close events greatly affects our emotions, although being banal conversations, these emotions will take on normality over time. Since our work focuses on short elevator conversations, and how emotions affect each other, we do not evaluate the memory or durability of these emotions. We only focus on how emotions alter the current conversation.

We can observe this in the following tables. In **Table 2** we see a template dialogue where the first character, A, has the emotions [joy: 3.6, anger: 1.4, sadness: 1.1, fear: 1.8, disgust: 1.3], while the character B has the emotions [joy: 1.2, anger: 1.4, sadness: 3.2, fear: 3.0, disgust: 1.3]. **Table 3** shows the result obtained from the same template swapping the emotions of the characters, where it is easy to see that the result of the dialogue is different. In addition, the reader can also see how the emotions of the character influence the response to the previous sentence.

Discussion

As a result of this work, we can emphasize that, although we are using a dictionary obtained from native speakers, there are words with confusing emotions, just as there is the absence of others that we consider emotional, like the days of the week. As long as this dictionary continues to evolve, the results provided by our system are likely to improve.

Among the points that we have not tackled is the analysis of sentences with negations or sarcastic meanings. These types of sentences need a different approach, since the identification of these factor require additional Natural Language Processing techniques. As with the use of sarcasm, there is a possibility that a character may feel envy. In these cases, when a person feels envy and tries to hide it, he resorts to lies, where the emotions that are perceived in the sentences are the opposite of those he really feels (false happiness).

In order to address all this, it is necessary to take into account the personalities of the characters, which not only helps us recognize selfish characters prone to envy, but there are also types of personalities where the emotional impact of a conversation is different: for example, an empathetic character tends to be more influenced in a conversation than an apathetic one.

Conclusion

The system we have presented is capable of generating short dialogues between two characters, with the content of an elevator conversation. These dialogues present, as their main characteristic, the influence of the emotions of the characters on the dialogue obtained in the end.

During the progress of the conversation, the emotions that the characters perceive from the sentences they receive directly affect their emotional state. This causes not only that the conversations take into account the isolated emotion of each of the characters, but the influence and domination that one type of emotions has over the others. This influence is due to the values of the emotions and their nature, since, as we have previously seen, not all the emotions have the same influence on each other.

Thereby, and taking into account the main objective of this article, which was to create elevator conversations based on the emotions of the characters and the influence they have on that conversation, we can state that the results are satisfactory. We managed to implement a system capable of generating dialogues where emotions are the drivers of the conversation, although we have identified that there is much room for improvement on colloquial expressions, sarcasm and personality of the characters.

Finally, from the two examples mentioned above, we can see in **Tables 4 and 5** two more examples of dialogues created with emotions.

Future Work

As we have seen in the Discussion section, there are many open lines of work for improvement. To highlight some, we believe it is important to address a better phrase analyzer where negation and colloquial expressions are addressed.

Another aspect that must be tackled is to include personality to the characters. In this way we could incorporate factors such as lying, sarcasm, empathy (emotional influence), interest in the other or self-centeredness.

As for the technical improvements, we believe that the next step is to incorporate other NLG techniques that allow us to overcome the limitations of the template-based systems. These systems are very good for rapid prototyping, but they are very expensive to maintain and not very scalable. We will analyze systems based on linguistic dictionaries and machine learning systems based on neural networks.

Finally, we aim to integrate our Emotional Dialogue Generation system as a component that can be used by other storytelling systems such as INES (Concepción, Gervás, and Méndez 2018), with the aim of nourishing and improving narrative generation by incorporating character dialogues.

Character	Generated in Spanish (Translated into English)	joy	anger	sadness	fear	disgust
A	Buenas tardes, vecina. Como esta hoy? (<i>Good afternoon, neighbor. How are you today?</i>)	3.37	2.33	1	1	1.9
B	Me duele la espalda, pero he salido al parque por la tarde. (<i>My back hurts, but I went to the park in the afternoon.</i>)	2.75	1	3.78	1.85	1.4
A	Naturalmente, una se aburre tantas horas aqui. (<i>It's normal, anyone gets bored for being so many hours here.</i>)	3.76	1	3.15	1	1
B	Que fastidio, ahora empieza a mejorar el clima. (<i>How annoying, now the weather begins to improve.</i>)	4.26	3.25	1	1.15	1.3
A	Disfruta! (<i>Enjoy!</i>)	4.45	1	1	1	1

Table 2: A=[joy: 3.6] - B=[sadness: 3.2, fear: 3.0]

Character	Generated in Spanish (Translated into English)	joy	anger	sadness	fear	disgust
A	Me duele la espalda, pero me he bajado al parque por el día. (<i>My back hurts, but I've gone down to the park for the day.</i>)	3.80	1	3.83	1.85	1
B	En fin, cualquiera se aburre tantas horas sin hacer nada. (<i>Anyway, anyone gets bored for being so many hours doing nothing.</i>)	3.18	1.73	3.92	1.15	1.90
A	Por suerte, ahora empieza a refrescar un poco y creo que se fastidiará la tarde. (<i>Luckily, now it starts to cool a bit and I think it will spoil the afternoon.</i>)	3.73	3.25	2.55	1.15	1.3
B	Adios. (<i>Bye</i>)	1	1	1	1	1

Table 3: A=[sadness: 3.2, fear: 3.0] - B=[joy: 3.6]

References

- Bollen, J.; Mao, H.; and Pepe, A. 2011. Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. *Icwsm* 11:450–453.
- Bolles, R. C., and Fanselow, M. S. 1980. A perceptual-defensive-recuperative model of fear and pain. *Behavioral and Brain Sciences* 3(2):291–301.
- Cavazza, M., and Charles, F. 2005. Dialogue Generation in Character-based Interactive Storytelling. In *Artificial Intelligence and Interactive Digital Entertainment*, 21–26.
- Concepción, E.; Gervás, P.; and Méndez, G. 2018. Ines: A reconstruction of the charade storytelling system using the afanasyev framework. In *Ninth International Conference on Computational Creativity, ICCO 2018*.
- Das, D., and Bandyopadhyay, S. 2009. Word to sentence level emotion tagging for bengali blogs. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, 149152. nullUSA: Association for Computational Linguistics.
- Delatorre, P.; Leon, C.; Gervas, P.; and Palomo-Duarte, M. 2017. A computational model of the cognitive impact of decorative elements on the perception of suspense. *Connection Science* 29(4):295–331.
- Eugercios, G.; Gutierrez, P.; and Kaloyanova, E. 2018. Analisis emocional para la inclusion digital. Bsc. thesis, Universidad Complutense de Madrid.
- Ferré, P.; Guasch, M.; Martínez-García, N.; Fraga, I.; and Hinojosa, J. A. 2017. Moved by words: Affective ratings for a set of 2,266 spanish words in five discrete emotion categories. *Behavior research methods* 49(3):1082–1094.
- Ghosh, S.; Chollet, M.; Laksana, E.; Morency, L.-P.; and Scherer, S. 2017. Affect-Im: A neural language model for customizable affective text generation. *arXiv preprint arXiv:1704.06851*.
- Hinojosa, J. A.; Martínez-García, N.; Villalba-García, C.; Fernández-Folgueiras, U.; Sánchez-Carmona, A.; Pozo, M. A.; and Montoro, P. R. 2016. Affective norms of 875 spanish words for five discrete emotional categories and two emotional dimensions. *Behavior research methods* 48(1):272–284.
- Lerner, J. S.; Li, Y.; Valdesolo, P.; and Kassam, K. S. 2015. Emotion and decision making. *Annual Review of Psychology* 66(1):799–823. PMID: 25251484.
- Mahamood, S., and Reiter, E. 2011. Generating affective natural language for parents of neonatal infants. In *Proc. of*

Character	Generated in Spanish (Translated into English)	joy	anger	sadness	fear	disgust
A	¿Que tal vecino, como va todo? (<i>What's up neighbor, how is everything going on?</i>)	2.58	1.65	1.85	1.86	1.91
B	Buenos días amigo. Pues mi madre viene por vacaciones (<i>Good morning friend. Well, my mother is coming for a vacation</i>)	3.82	1	1	1	1
A	Fantástico, me alegro mucho, disfruta los buenos momentos (<i>Fantastic, I'm glad, enjoy the good times</i>)	4.73	1	1	1	1
B	Se agradece, un saludo (<i>Thank you, best regards</i>)	4.39	1	1	1	1

Table 4: A=[joy: 2.6] - B=[joy: 2.2, disgust: 2.3]

Character	Generated in Spanish (Translated into English)	joy	anger	sadness	fear	disgust
A	Que tal, ¿sabes cuando empieza el partido hoy? (<i>How are you, do you know when the game starts today?</i>)	4.76	1.43	1.3525	1.47	1.37
B	Buenas, empieza a las cinco y media. (<i>Hi, it starts at five thirty.</i>)	3.35	1	1.6	1	1
A	Genial me iré un poco más pronto con la idea de intentar llegar a tiempo. (<i>Great I'll leave a little sooner with the idea of trying to arrive on time.</i>)	3.173	1	2.55	1	1
B	Te dará tiempo. (<i>You will be on time.</i>)	3.3	1	1	1	1

Table 5: A=[joy: 4.2] - B=[sadness: 3.6]

the 13th European Workshop on Natural Language Generation, 12–21. Association for Computational Linguistics.

Marsella, S.; Gratch, J.; Petta, P.; et al. 2010. Computational models of emotion. *A Blueprint for Affective Computing-A sourcebook and manual* 11(1):21–46.

Méndez, G.; Gervás, P.; and León, C. 2016. *On the Use of Character Affinities for Story Plot Generation*, volume 416 of *Advances in Intelligent Systems and Computing*. Springer. chapter 15, 211–225.

Minsky, M. 2010. La máquina de las emociones. *Del dolor al sufrimiento* 3:92–125.

Miranda, C. N. H.; Luna, J. A. G.; and Morillo, D. S. 2016. Minería de opiniones basado en la adaptación al español de anew sobre opiniones acerca de hoteles. *Procesamiento del Lenguaje Natural* 56:25–32.

Oh, A. H., and Rudnicky, A. I. 2000. Stochastic language generation for spoken dialogue systems. In *Proc. 2000 ANLP/NAACL Workshop on Conversational systems-Volume 3*, 27–32. Assoc. for Computational Linguistics.

Oñate, A.; Méndez, G.; and Gervás, P. 2019. Introducing mood and affinity to generate brief template-based dialogues in storytelling systems. In *C3GI: The 7th International Workshop on Computational Creativity, Concept Invention, and General Intelligence*. Bozen-Bolzano, Italy: CEUR Workshop Proceedings.

Pak, A., and Paroubek, P. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, 1320–1326.

Polzin, T. S., and Waibel, A. 2000. Emotion-sensitive human-computer interfaces. In *ISCA tutorial and research workshop (ITRW) on speech and emotion*.

Quan, C., and Ren, F. 2010. Sentence emotion analysis and recognition based on emotion words using ren-cccps. *International Journal of Advanced Intelligence* 2(1):105–117.

Shawar, B. A., and Atwell, E. 2007. Different measurements metrics to evaluate a chatbot system. In *Proceedings of the workshop on bridging the gap: Academic and industrial research in dialog technologies*, 89–96. Association for Computational Linguistics.

Stadthagen-González, H.; Ferré, P.; Pérez-Sánchez, M. A.; Imbault, C.; and Hinojosa, J. A. 2018. Norms for 10,491 spanish words for five discrete emotions: Happiness, disgust, anger, fear, and sadness. *Behavior research methods* 50(5):1943–1952.

Traum, D. 2017. Computational approaches to dialogue. *The Routledge Handbook of Language and Dialogue* 143.

Velsquez, J. 1997. Modeling emotions and other motivations in synthetic agents. *Aaai/iaai* 10–15.

Wang, W.; Chen, L.; Thirunarayan, K.; and Sheth, A. P. 2012. Harnessing twitter” big data” for automatic emotion identification. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, 587–592. IEEE.

Zhou, H.; Huang, M.; Zhang, T.; Zhu, X.; and Liu, B. 2018. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

MASSE: A System for Music Action Selection Through State Evaluation

Prashanth Thattai Ravikumar

Department of Communications and New Media
National University of Singapore
prashanth.thattai@u.nus.edu

Lonce Wyse

Department of Communications and New Media
National University of Singapore
cnmwill@nus.edu.sg

Abstract

This paper concerns the development of a music co-creation system that engenders creative partnerships in musical interaction. Towards this goal, specific design guidelines are identified for improving the musicians' sense of creative partnership with the system through behaviors that communicate musical intent, alternate musical roles based on musical stability, and negotiate transitions in musical characteristics based on togetherness in interaction. The design guidelines motivate the development of an agent architecture for interactive improvisation. The architecture enables an agent to select music response behaviors based on the stability and togetherness sensed during interaction. An agent improvisation system is developed that uses the architecture to make decisions during improvised percussion duets. The agent negotiates transitions between musical roles and musical characteristics by maintaining different degrees of metrical coherence and similarity with the rhythms played by the musician. By negotiating transitions with a musician, the system presumably has an impact on the musicians sense of creative musical partnership.

Introduction

The central question that is of interest here in this work is - how do we design systems that are creative musical partners to human musicians during live interaction? Free improvising musicians identify three functions that are important to engender a sense of creative partnership (or *co-experience*), namely - achieving communication, alternating musical roles, and negotiating transitions (Ravikumar, McGee, and Wyse 2018). In this paper, we discuss the design of a music co-creation system that performs the above mentioned functions to engender a sense of co-experience during real-time interaction with an improvising musician.

In order to study the impact of system design on co-experiences, we review work on music generation systems that are designed to perform alongside human musicians. The subset of music generation systems that are of interest here utilize sound as the primary medium of interaction with the musician. In the rest of the paper, these are termed as music response systems. Other systems have been developed that use note-based representations (Pachet 2003; Assayag et al. 2006) for direct engagement, and extra-musical

(François, Chew, and Thurmond 2007; Cicconet, Bretan, and Weinberg 2012), and notational devices (Rowe 2004; Nika and Chemillier 2012) for making decisions along with the musicians. In group free improvisation situations that are of interest in this work, musicians often free themselves from the restrictions of note based interactions and focus on interacting through the sounds that are produced by the performers. During their interaction, musicians also report that extra-musical communication such as eye contact and physical gesturing is often a distraction from their interaction through sound, and prefer to play along by listening to the sounds of other musicians (Ravikumar, McGee, and Wyse 2018). The rest of the paper is organized around the issues and challenges in engendering a sense of co-experience with music response systems.

In the following section, specific examples of music response systems are presented to highlight the impact of designing specific system components on the musicians' sense of co-experience with the system.

Music response systems

Related work has focused on developing music response systems that interact using musical actions, alternate between lead and follower roles, and make decisions to negotiate transitions in the music. Reports are gathered from musicians' interactions with various systems and the impact of each component on the musicians' sense of co-experience is analyzed. Following the analysis, design guidelines are proposed to improve each component of the system.

Interaction through musical actions

Music response systems interact through musical actions that respond, in varying degrees, to the musical characteristics identified in the musicians' input. We review two music response systems that use different action descriptions. In the first system, musical actions are designed to produce musical material that is related to the sound made by the musician (Brown, Gifford, and Voltz 2016). In the second work, system uses musical actions that are intended to communicate a symbolic response (e.g., agreement, disagreement) to the musician along with the content (Murray-Rust and Smaill 2011). In this work, sonic interaction with a music response system is improved through musical actions that communicate the systems' intent to the musician.

Brown and his colleagues developed a music improvisation system that triggers musical actions to interact with the musician (Brown, Gifford, and Voltz 2016). The system contains a collection of six interaction behaviors - *repeat, imitate, shadow, initiate, silence, and turn taking* that it uses to respond to a musician. During the performance, the system triggers the behaviors at random. In a study that involved duet interactions with the system, the authors analyzed the impact of the system on musicians' co-experiences. Musicians who played with the system reported that they felt most engaged with the system during the duet interaction. However, they also felt a lack of creative initiative on the part of the computational co-creator.

The second system uses symbolic musical actions that are associated with communicative meaning (e.g., agree, disagree) to interact with the musician (Murray-Rust and Smaill 2011). During the performance, the system selects a musical action using a two step procedure. First, it listens to interpret the relation between the musical material played by the system and the musician and represents them as symbolic actions. Then, it consults a corpus to retrieve the continuation of the current action. Musicians interacted with two different system configurations, one that constructed musical actions through retrieval mechanism, and one that always imitates the musician. While musicians reported that their sense of interactivity was marginally higher with the system configuration that constructed musical actions through retrieval from the corpus, there were no significant differences to interactivity across the different conditions.

In summary, musical actions that are used in these music response systems are effective in engendering responsive interaction (e.g., initiating music), but they are restricted in their ability to communicate musical intent. Canonne and Garnier (2015) observe that musicians use strategies that combine their actions with the actions of other musicians. These strategies involve stabilizing their playing, waiting to see whether the other musician is going along, playing along, or densifying the musical texture. Musicians use these strategies to clearly communicate their intent.

In order to improve the ability of the system to impact co-experiences, we propose the design of music response behaviors that combine with the actions of the musicians to communicate musical intent. This is achieved by designing behaviors that stabilize through repetition, wait to let the musician play, play along with the current musical characteristics, and densify the musical texture of the improvisation.

Alternating roles

Some music response systems maintain or change their response behavior in unpredictable ways in order to introduce changes in musical roles. In particular, musicians are often unaware of the exact time at which the system introduces changes in its behavior. In order to respond to behaviors that are unpredictable, musicians change their degree of attention and responsiveness to the system's actions, and consequently their role with respect to the system. Here, we look at three systems that use different processes to achieve this

effect. The first system changes its behavior by changing the musical source that it listens to and uses for response generation (Lewis 2000). Another system changes behavior by altering the variability in its responses with respect to the musicians' input and inbuilt internal goals (Young, Bown, and others 2010). The third system alters its behavior through changes in its internal state (Donnarumma 2017). While the above mentioned systems bring about a change in the musical roles, the roles change because the musician adapts to the changes introduced by the system.

The Voyager system is notable for being recognized by several authors and musicians as an equal co-improviser (Lewis 2000). During the performance, the system changes behavior by changing the musical sources that it listens to and the number of musical channels that it uses for output. These changes are triggered through chance processes that musicians will not be able to predict beforehand. Following the change, the system resumes its usual responsive behavior that involves adapting to the timbre and textural elements to the input from the changed source. The responsive behavior of the system engenders a feeling that the system has a personality of its own. However, the unpredictability in the system's behavior often makes it difficult for the musician to follow the changes.

The second system (Young, Bown, and others 2010) introduces changes based on the musicians' input and an internal goal state. Young, Bown, and others designed a system that improvises by clapping rhythmic patterns along with a musician. The system responds by generating variations on the last rhythm played by the musician, and selecting the variations that are most similar to a target rhythm. In certain configurations when musicians played with low variability, they were able to interpret the changes that were introduced by the system. But the system's behavior was found to be erratic and difficult to play along with when musicians' varied their rhythm patterns. As a result, musicians found it difficult to engage with the unpredictable behaviors exhibited by the system.

The third variety of systems introduce variability in their behavior through autonomous state changes. One such system designed by Donnarumma (2017) gets input from the performers' body gestures and generates musical responses based on its dynamically evolving state. The performer restricts the scope of actions in order to interpret the changes made by the system. With these restrictions in place, the performer is able to interpret the behaviors initiated by the system, such as a crescendo, as a part of musical movement. While imposing restrictions appears to be effective in working along with the unpredictable nature of the system, performing with these constraints requires considerable training on the part of the human performer.

In summary, these three musical systems behave unpredictably in order to bring about changes in musical roles, but the changes in the roles are a result of the musician adapting to the system's unpredictability. In human improvised performances, co-improvising musicians make decisions to alternate roles based on the degree of relative stability that they maintain with respect to the other musicians during the interaction (Canonne and Garnier 2012). When musicians

sense that the group has been playing with too few changes, they introduce instability by diversifying the musical material. When musicians notice that there are too many changes, they introduce stability in the music (e.g., through repetition).

Improvements to the design of systems such as those described above require the development of modules that introduce changes in roles by sensing the relative stability of the systems' behavior with respect to behavior of the musician in the interaction.

Negotiating transitions

Music response systems negotiate transitions along with the musician by changing their musical configuration over a period of time. One such system changes its musical configuration through incremental adjustments to the music generation parameters (Albert 2013). Another system responds to the differences sensed in low level parameters with rapid changes in the musical variables (Bown 2018). As an improvement in design, the system makes decisions to change its musical characteristics by sensing its state of togetherness with the musician. The specific design guideline for developing this aspect of a music response system involves the development of a listening component that monitors the degree of togetherness in its actions with respect to the musician.

Let us consider the first system that introduces changes to its musical configuration independently of the musician (Albert 2013). The system contains a collection of music generation parameters it varies during the interaction. In one of the experimental configuration of the system, the systems introduces gradual and incremental changes to the music generation parameters in order to change the musical characteristics. Musicians who played with the system reported that the changes introduced by the system were responsive to their playing. However, musicians also felt a "latency between the human's action and system's adoption of that action" (when IMP chooses to adopt that action) (Albert 2013).

Bown's (2018) system, on the other hand, is designed to be very responsive to the slightest changes introduced by the musicians. The systems' immediate responsiveness to the low level parameters and their non-linear coupling with higher level musical behaviors introduces rapid changes in the music. While it is clear that the system adopts to the changes introduced by other performers, musicians face a different challenge with the system. During the interaction, the system introduces rapid changes that are difficult to follow. In these moments, the musicians feel the need for the system to slow down, and to "lock in to the moment".

Existing musical systems perform transitions with an improvising musician, but are unable to negotiate them in a manner that feels interactive. Interaction during transitions is an important characteristic of collective free improvisation as there is a high degree of correlation between the decisions made by the musicians about the occurrence of the transition points (Canonne and Garnier 2015). Wilson and MacDonald (2016) found that during the moments of transitions, human musicians make decisions to maintain or change their ac-

tions by evaluating their togetherness with co-improvising musicians (Wilson and MacDonald 2016). In particular, human musicians seem to evaluate the degree of togetherness through the homogeneity of the combined musical output. When musicians sense that they have been playing musical material that is homogeneous for a long time, they perform strategies that diversify the musical texture. Similarly, when musicians sense that they are playing material that is very complementary to each other, they adopt strategies that bring the musical material closer to each other.

Improvements to the design of systems such as those described above require the development of modules that negotiate transitions based on sensing the togetherness in the musical output that is jointly produced by all the musicians.

In summary, there are two aspects of related work that offer scope for improvement based on the strategies used by free improvising musicians. The ability of the system to respond to musician's actions is improved through the design of musical actions that combine with the actions of other musicians to communicate musical intent. The decision making strategies that are used by the system are improved by evaluating the relative stability and togetherness of the musical output to make decisions. Next, the different design recommendations that have emerged from the literature described above are integrated within a new agent architecture.

Agent architecture

The agent's architecture is divided into two parts: 1) a reactive part, and 2) a decision making part. The reactive part contains music response behaviors that agent uses to generate music from input. The decision making part stores the sequences that it listens to, and evaluates based on stability and togetherness. During interaction, the agent generates music by choosing musical actions that combine with the musical sequences played by the musician. The agent also makes decisions to maintain or change its selected actions by evaluating the degree of togetherness, and stability during its interaction with the musician.

Musical action

The agent uses musical actions to generate sequences that are responsive to the input from the musician. Musical actions are transformation functions that combine two or more musical sequences to produce a third sequence that is partially similar and partially different from the input sequences. Each musical action that is used in the agent requires two musical sequences as input - a sequence that is internally generated by the system, and a sequence that is obtained from a musician or from the environment. New sequences are generated by combining the musical characteristics of the internal sequence with the characteristics of input.

Action descriptions are central to the reactive part of the system labeled as region (a) in Figure 1. The reactive part of the system is responsible for the generating musical material that is responsive to the real-time input from the musician.

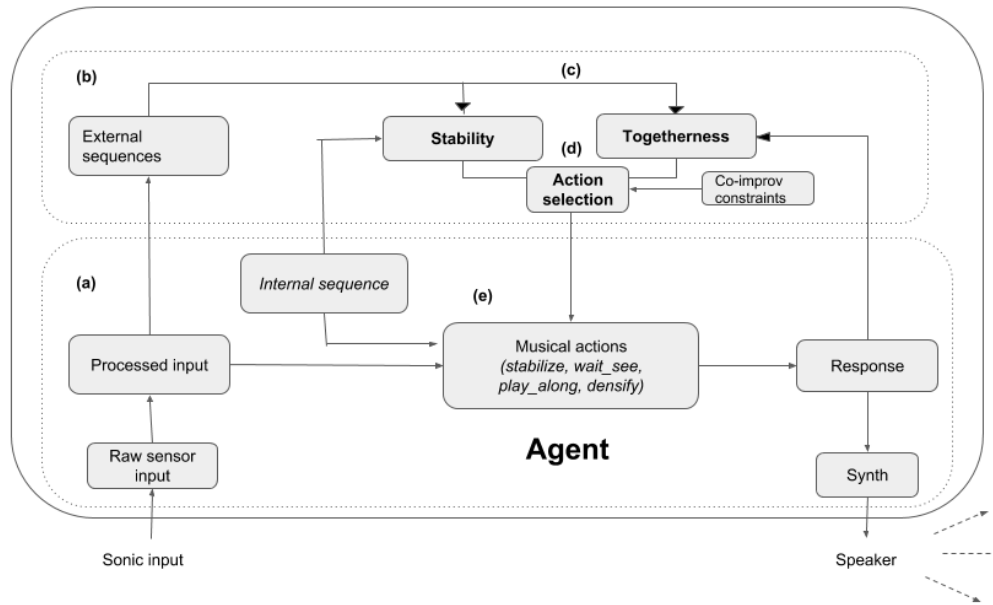


Figure 1: Parts of the agent architecture: *a) Reactive part, b) State and decision making, c) Scoring states, d) Selecting new action, e) Generating musical material*

The reactive part contains modules for music generation and corresponds to the region (e) of Figure 1. Once an action is selected, the two sequences - an internal sequence and an input sequence are combined to generate new musical material.

Strategy	Description
Stabilize	Repeat source pattern
Wait and see	Play only when musician plays
Play along	Imitate the characteristics of musician
Densification	Introduce characteristics different from self and past

Table 1: Music response behaviors

During the interaction, the agent uses the four actions (listed in Table 1) to generate musical material that is responsive to the input from the musician. The agent uses the generated musical material to stabilize the music, wait to see where the musician is taking the music, play along with the musician, or densify the playing with respect to the musician. The descriptions of the four actions are presented in Table 1.

The musical actions are implemented through operations that combine musical characteristics from the internal sequence and the input sequence. As an example, one possible set of operations that combine the intensity levels of the internal sequence and the input sequence is described here. In order to produce new sequences that maintain stability in the music, the agent assigns an intensity level that is equal to the mean intensity level of the internal sequence and cor-

responding input from the musician. To implement the wait and see strategy, the system selects events from the internal sequence only when the intensity of the input exceeds a threshold. In all the other conditions, the agent remains silent or selects an intensity value of 0. To play along with the musician, the agent decreases the intensity of its output sequence when it senses that the intensity of input is high and increases the intensity when the input is low. In order to densify the musical texture, the system increases the intensity of its output when it senses that the input is high, and maintains its intensity when the input is low.

The agent performs the above mentioned operations at the rate that is equal to the rate at which it receives inputs (every 20 ms). Every input that is received from the musician is interpreted, and combined with an internal event to produce a new musical event. The ability to react to musical changes at a short time scales keeps the agent responsive to sudden changes in the input during interaction.

The remaining parts of the agent architecture (region (b) of Figure 1) use state to make decisions and select actions and are described here.

State

In order to make decisions, the agent evaluates the improvisation by keeping track of two metrics of its interaction with the musician, namely, togetherness and stability. This information is represented as numerical values that correspond to the degree of togetherness and the degree of stability. The numerical values are computed using the features of the musical sequences that the agent has listened to and played in the past.

The agent represents the information about togetherness and stability through numerical values between 0 and 1. A high degree (1) of togetherness corresponds to a situation in which musicians play musical material that is very similar to other musicians, and a high degree of stability (1) corresponds to a situation in which the agent plays musical material that maintains musical coherence with the musician. The values assigned to togetherness and stability are stored in state variables that the agent uses for decision making.

The procedure for assigning a numerical score to togetherness and stability is explained through notions of internal and external component of the state (region (c) in Figure 1). The internal component of the state keeps track of the musical material that is generated by the agent (e.g., internal sequences, responses). The external state stores the musical material that the agent receives as input. It is important to note that unlike the processes of response generation, state is computed at a timescale on the order of 5 seconds. During the period of listening, the agent pairs the input from the musician and the internal sequence to form musical sequences. Then, it computes the differences in the musical sequences that are stored in the internal and the external state to compute the degree of togetherness and stability. The different steps that the agent follows to assign numerical scores to the state are described in detail below.

Scoring togetherness The agent scores the degree of togetherness by scoring the differences between the responses generated by the agent and the external sequences stored in the external state. The following procedure is used to score the differences that the agent senses between the musical material.

- The agent continuously obtains new sonic input from the environment in terms of musical features (e.g., intensity values, spectral changes). The musical features are directly obtained from the sensor and stored as musical sequences. The agent generates an internal sequence that it combines with the input sequence.
- The musical features are concatenated to form musical sequences that are stored in memory. These sequences are stored in the external state variable.
- While the agent is listening to input, it is also generating new musical material using the process described in region (e). The agent stores the generated responses in an internal state variable.
- In order to compute togetherness, the agent compares the values of the musical sequences in the external state with the responses stored in the internal state and assigns a numeric value based on the magnitude of the differences.
- The numeric value of differences is converted to ratio that corresponds to the number of differences per unit of time (e.g., beat)
- Finally, the numeric value of differences is subtracted from 1 to obtain the degree of togetherness.

$$degree_{togetherness} = 1 - ratio_{difference} \quad (1)$$

The highest value of degree of togetherness is 1 and corresponds to a situation when musicians are playing musical

material that are completely homogeneous with respect to their changes in intensity and timbre values. This signifies the state of togetherness in playing. The lowest value of togetherness is 0, and corresponds to a situation when musicians are playing musical material that is complementary or contrasting with respect to the changes in intensity and timbre values. This signifies a state of indifference between the agent and the human. Values that lie between 0 and 1 correspond to different degrees of togetherness.

Scoring stability The agent assigns a numeric value to the stability in the interaction based on the correlation between the musical sequences expected by the agent and the actual musical sequences played by the musician. The first 3 steps in the procedure for scoring stability are the same as the steps for scoring togetherness.

- Step (1), (2), and (3) from scoring togetherness.
- The agent computes the correlations between the musical features that are stored in the internal and external state.
- The correlation values are converted to a range between 0 and 1 to correspond to the degree of stability.

The highest value of degree of stability is 1 and corresponds to a situation when the agent finds a strong correlation between the events that it expected to occur and the events that the musician played in the performance. The lowest value of stability is 0, and corresponds to a situation when the agent finds a weak correlation between the events that it expected to occur and the events that the musician played in the performance. Values that lie between 0 and 1 correspond to different degrees of stability in the interaction.

Scoring co-improvisation state The values that were assigned to the degree of togetherness and the degree of stability are added to assign a numerical value to the state of co-improvisation.

$$coimprov_{score} = degree_{togetherness} + degree_{stability} \quad (2)$$

Next, the musical actions and the state values are integrated via a mechanism that selects musical actions based on state information.

Action selection

Using the co-improvisation score, the agent makes decisions to maintain or change its actions during live interaction (region (d) of Figure 1). First, the agent assigns a numerical score to the co-improvisation state by computing the degree of togetherness and degree of stability. Then, it compares the co-improvisation score against a target score to check whether the current state satisfies the internally specified constraints of co-improvisation. When the agent determines that the state score does not meet constraints, it selects a new state to move to, and selects one of the available musical actions to reflect its decision to change. Given that the agent does not know the sequences that the musician will play, it makes decisions to maintain or change actions based on the conditional assumption that musicians will repeat their action.

Decision boundaries Decisions boundaries are constructed in order to enable the agent to make decisions to maintain or change its state. The decisions boundaries divide the state space into ranges that correspond to different regions of togetherness and stability with another musician. Periods of high stability and high togetherness are characterized by values that are between 0.75 and 1. Values between 0 to 0.25 correspond to regions of playing in which musicians play with low stability (high variability) and low togetherness. Values between 0.25 and 0.75 correspond to the periods of playing in which musicians balance stability with togetherness in order to play along with other musicians. Co-improvisation values that lie in the range of 0.25 to 0.75 are ideal for the agent to play along with the musician.

Decision to maintain/change state The numerical co-improvisation score that is computed in the previous steps is compared with the decision boundary values to make decisions to maintain or change the state. The agent maintains its current state when it senses that the state score is within the acceptable bounds. The agent changes its state when the state score lies outside the acceptable bounds. The agent implements the change by selecting a new action from the available list of actions.

Selecting new action In order select a new action, the agent scores the stability values for each of the alternate actions. The stability scores are computed using the processes described in previous steps, and combined with the degree of togetherness score to generate a new co-improvisation score. The new co-improvisation score is checked with decision boundary values to determine whether the alternate actions that are available to the agent are viable. The agent selects one of the alternate actions that is produces a score closest to the desired co-improvisation state. In the event that the agent finds that no actions bring the co-improvisation score within bounds, the agent selects one of the alternate actions at random. The selected action is combined with the musical input to produce musical material.

Rhythmic improvisation agent

The architecture is used to develop a rhythmic improvisation system that selects musical actions to perform rhythmic duets. The various components of the agent architecture that are developed for interactive rhythmic improvisation are described here.

Numeric scoring functions The agent improvisation system that is developed in this work computes numerical measures of stability and togetherness using the notions from Gifford and Brown (2006) as well as Cao, Lotstein, and Johnson-Laird (2014).

Gifford and Brown implemented an interactive rhythm improvisation system that generated rhythmic accompaniment that varies in its degree of coherence with respect to a rhythmic input (Gifford and Brown 2006). Rhythmic

coherence was numerically measured through the correlations between the meter, velocity, pitch, and the duration of events in two rhythm patterns (Gifford and Brown 2006). The lower the rhythmic coherence between the rhythms, the less stability that musician feels while playing with the agent. The higher the rhythmic coherence between the rhythms, the more stability that musician feels while playing along with the agent.

Cao, Lotstein, and Johnson-Laird (2006) propose a numeric metric of rhythmic similarity based on the similarities in metrical organization of rhythm patterns. The agent that is developed in this work assigns a numeric value to togetherness based on the similarities in three metrical units - syncopated notes, notes on the musical beat, and other events (e.g., rests) - that it finds with the musician.

Rhythm representation In order to assign numerical values to stability and togetherness using the notions from Gifford and Brown as well as Cao, Lotstein, and Johnson-Laird, the agent uses a three part representation of rhythm patterns. Each rhythm pattern is specified through three parameters: 1) binary sequences of hits and rests for 16 hits, 2) intensities that correspond to each hit in the binary sequence, 3) the timbre that is played at each hit. Each hit in the source rhythm has a duration of 1/8th beats at the tempo of 60 beats per minute.

Musical actions Using the rhythm representation mentioned above, the improvisation agent chooses actions that alter the intensity and the metrical division of a musical unit (e.g., a beat). Alterations to intensity and metrical subdivisions changes the density and syncopation of the rhythm, and subsequently affects the metrical stability and rhythmic similarity values. Each musical action generates an internal sequence with metrical subdivisions for each beat (e.g., single hits, eight notes). In order to *wait and see* where the musician is going, the agent generates internal sequences with a musical meter that has a lower metrical subdivision compared to the previous meters played by the musician and the agent. To *play along* with the musician, the agent generates internal sequences that have the same number of metrical subdivision as the musician. The *stabilization* action repeats sequences with the same metrical division that the agent is currently playing with. In order to *densify*, the agent generates internal sequences with musical meters that have a higher number of metrical subdivisions compared to the previous meters played by the musician and the agent.

Computing numerical values After each action, the agent computes the numerical scores of similarity and stability to compute the co-improvisation score. Each 1/8th division of the beat is scored based on whether the agent and the musicians registered a hit or left a rest at that beat division. If musicians performed a hit in the place of a rest left by the agent on a 1/8th beat, or vice versa, it counted as a single unit of difference. Togetherness is the ratio of number of differences in 16 1/8th notes. The degree of togetherness is computed as *1-ratio of togetherness*. In order to measure metrical stability, the agent

finds the correlations between velocity and duration of hits played by the musician and the agent. Timbre values are not used in stability evaluation. Differences in timbres are experimentally controlled and do not contribute to metrical stability. Based on the numerical scores that are computed after each action, the agent computes the score for the co-improvisation state. The sum of togetherness and stability is normalized to 1 and corresponds to the co-improvisation score.

Selecting actions The agent compares its co-improvisation score with the boundary values to maintain or change its musical action. When the agent determines that the co-improvisation score is higher or lower than the range, it changes its behavior. When the agent determines that the co-improvisation score is within the range, it maintains its behavior. The boundary values were experimentally set to 0.25 and 0.75 as they produced to the periods of interaction in which agent balanced stability with togetherness.

Performing with the system An improvisation situation in which the agent negotiates transitions along with the musician is described here. Through an unpredictable change, the agent negotiates a change in musical texture and brings the co-improvisation value within bounds. At the beginning of the interaction, the agent selects a musical action and repeats it. Once the musician begins to play, the agent senses an increase in the degree of togetherness and stability in the performance. The agent maintains its action until the value of the co-improvisation state is within the range specified by the decision boundaries. When the agent determines that the value assigned to the state is outside the range, it chooses a new musical action that either introduces a greater (*densify*) or lesser (*wait and see*) metrical subdivision of the beat with respect to the musician. By maintaining and changing the metrical subdivisions in the beat, the agent introduces changes in the musical texture while keeping the co-improvisation score within bounds. Videos of performances with the agent improvisation system are available at <https://prashanthiccc19.wordpress.com>.

Discussion

In this paper, state-of-the-art music response systems were evaluated to derive two design guidelines that were used to develop an agent architecture for music co-creation. A rhythmic improvisation agent was developed based on the architecture that plays along with musicians. During the interaction, the agent negotiates transitions between regions that vary in the degree of metrical coherence and similarity to the rhythms played by the musician. An agent was developed from the design guidelines that generates music by combining its musical actions with the actions of the musician. While earlier systems select specific actions that communicate an intent to the musician (Murray-Rust and Smaill 2011), the agent that is developed in this work combines its actions with the actions of

the musician to communicate its intent. It remains to be studied whether an agent that communicates its intent by combining its musical actions with the musician's actions improves the musicians' co-experiences with the system.

During moments of change, the agent makes decisions to maintain or change its response behavior by monitoring the changes in the co-improvisation state. With earlier systems, musicians have been unable to negotiate these moments in a manner that feels interactive. The particular procedure for decision making using co-improvisation state and the decision boundaries allows us to adjust the latency in the systems' responses to a range that feels interactive.

In order to compute the co-improvisation state, the agent monitors the togetherness and stability of its actions with respect to the musician. The operationalization of co-improvisation states through togetherness and stability allows humans to explain the decisions made by the system at an intuitive level of abstraction. A review by Karimi (2018) identified the need for improving the explainability of the decisions made by the co-creative systems that are otherwise non-interpretable. The ability to make decisions that are interpretable is also a first step towards improving the explainability of co-creative systems.

Finally, the rhythmic improvisation system that was implemented using the architecture measured stability through metrical coherence, and togetherness using rhythmic similarity. During the performance, the system negotiates transitions between different regions that vary in the degree of metrical coherence and similarity to the rhythms played by the musician. During the preliminary trials with the system, it was observed that the systems' behavior did not consistently correspond to the musicians' sense of changes in metrical coherence and similarity. A possible explanation is that the particular method of additively combining the scores of metrical coherence and similarity may not be an accurate measure of human judgement. Improvements to the system design will use metrics that have a closer perceptual correspondence with musicians' notions of togetherness and stability.

Conclusion

This paper concerns the design of a music co-creation system that engenders a sense of creative partnership. Towards this goal, specific design guidelines are identified to improve individual system components. The various components are integrated within an architecture to develop an agent that interacts by communicating musical intent, alters behaviors through monitoring the stability in interaction, and negotiates transitions by sensing togetherness in musical characteristics. In rhythmic improvisation situations, the agent improvisation system negotiates transitions in roles and musical characteristics through trade-offs between stability and togetherness. Future work will study the ability of the agent improvisation system to impact co-experiences.

Acknowledgments

I would like to thank Dirk Stromberg and Isaiah Koh for their insightful discussions.

References

- Albert, J. 2013. Interactive musical partner: A demonstration of musical personality settings for influencing the behavior of an interactive musical generation system. In *Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*.
- Assayag, G.; Bloch, G.; Chemillier, M.; Cont, A.; and Dubnov, S. 2006. Omax brothers: a dynamic topology of agents for improvisation learning. In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, 125–132. ACM.
- Bown, O. 2018. Performer interaction and expectation with live algorithms: Experiences with zamyatin. *Digital Creativity* 29(1):37–50.
- Brown, A. R.; Gifford, T.; and Voltz, B. 2016. Stimulating creative partnerships in human-agent musical interaction. *Computers in Entertainment (CIE)* 14(2):5.
- Canonne, C., and Garnier, N. B. 2012. Cognition and segmentation in collective free improvisation: An exploratory study. In *Proceedings of the 12th International Conference on Music Perception and Cognition and 8th Triennial Conference of the European Society for the Cognitive Sciences of Music*, 197–204.
- Canonne, C., and Garnier, N. 2015. Individual decisions and perceived form in collective free improvisation. *Journal of New Music Research* 44(2):145–167.
- Cao, E.; Lotstein, M.; and Johnson-Laird, P. N. 2014. Similarity and families of musical rhythms. *Music Perception: An Interdisciplinary Journal* 31(5):444–469.
- Cicconet, M.; Bretan, M.; and Weinberg, G. 2012. Visual cues-based anticipation for percussionist-robot interaction. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction (HRI'12)*, 117–118. Boston, Massachusetts, USA: ACM.
- Donnarumma, M. 2017. Beyond the cyborg: Performance, attunement and autonomous computation. *International Journal of Performance Arts and Digital Media* 1–15.
- François, A. R.; Chew, E.; and Thurmond, D. 2007. Mimi—a musical improvisation system that provides visual feedback to the performer. Technical Report 07-889, University of Southern California Computer Science Department.
- Gifford, T. M., and Brown, A. R. 2006. The ambidrum: Automated rhythmic improvisation. In *Proceedings of the Australasian Computer Music Conference*, 44–49. Adelaide, Australia: Australasian Computer Music Association.
- Karimi, P.; Grace, K.; Maher, M. L.; and Davis, N. 2018. Evaluating creativity in computational co-creative systems. *arXiv preprint arXiv:1807.09886*.
- Lewis, G. E. 2000. Too many notes: Computers, complexity and culture in voyager. *Leonardo Music Journal* 10(nil):33–39.
- Murray-Rust, D., and Smaill, A. 2011. Towards a model of musical interaction and communication. *Artificial Intelligence* 175(9):1697–1721.
- Nika, J., and Chemillier, M. 2012. Improtek: integrating harmonic controls into improvisation in the filiation of omax. In *Proceedings of the International Computer Music Conference (ICMC'12)*, 180–187.
- Pachet, F. 2003. The continuator: Musical interaction with style. *Journal of New Music Research* 32(3):333–341.
- Ravikumar, P. T.; McGee, K.; and Wyse, L. 2018. Back to the experiences: empirically grounding the development of musical co-creative partners in co-experiences. In *6th International Workshop on Musical Metacreation. 9th International Conference on Computational Creativity, ICCO*, 1–7.
- Rowe, R. 2004. *Machine musicianship*. MIT press.
- Wilson, G. B., and MacDonald, R. A. 2016. Musical choices during group free improvisation: a qualitative psychological investigation. *Psychology of Music* 44(5):1029–1043.
- Young, M. W.; Bown, O.; et al. 2010. Clap-along: A negotiation strategy for creative musical interaction with computational systems. In *Proceedings of the International Conference on Computational Creativity 2010*, 215–222. Lisbon, Portugal: Department of Informatics Engineering University of Coimbra.

Affordance-based Generation of Pretend Object Interaction Variants For Human-Computer Improvisational Theater

Mikhail Jacob and Prabhav Chawla and Lauren Douglas and Ziming He and

Jason Lee and Tanuja Sawant and Brian Magerko

Georgia Institute of Technology
Atlanta, GA 30308 USA

{mikhail.jacob, pc_1998, ldouglas7, zhe66, jlee3331, tanuja.sawant, magerko}@gatech.edu

Abstract

This paper describes DeepIMAGINATION, a neural architecture to generate variants of movement-based object interactions with props using the physical attributes of props while playing the Props game. The agent can generate these action variants while searching a learned action space in real-time to provide improvised responses to its human partner. Convolutional and recurrent variants of CVAEs are used for experimentation. The paper presents an evaluation of the architecture by benchmarking its ability to learn the human data set and generate believable, recognizable, and high-quality action variants from it. Results showed that the agent could generate believable, high-quality action variants. but that recognizability requires improvement.

Introduction

Human-agent improvisation is a challenging subset of human-computer co-creativity (mixed-initiative creativity) that requires improvisational agents to generate creative acts in near real-time within open-ended scenarios. The constraints of the task enforce severe temporal constraints on the agent while requiring the agent to possess a large amount of knowledge and reason about large action spaces without well-specified pre-defined goals at any given time. Constrained improvisational agents have been demonstrated in domains such as musical improvisation (Hoffman and Weinberg 2010), visual art (Davis et al. 2016), theater (Mathewson and Mirowski 2017), and dance (Reidsma et al. 2006); however, there are as yet few systems that focus on truly open-ended improvisation.

The authors' prior work investigated highly-constrained improvisation within theater (O'Neill et al. 2011) and pretend play (Magerko et al. 2014). Two problems became apparent from this initial work. Agents required a large amount of knowledge to be authored before they could respond meaningfully to a person's comparatively vast experiences and knowledge, also known as the knowledge-authoring bottleneck (Spierling and Szilas 2009). Previous work addressed this issue in the LuminAI installation (Jacob and Magerko 2015). It was also challenging for improvisational agents to perform meaningful real-time action selection from open-ended action spaces with ill-defined goals using learned embodied knowledge. This is the *improvisational action selection problem* that motivates this research.



Figure 1: Two actors playing the Props game from the popular TV show, "Whose Line Is It Anyway?".

A high-level solution to the improvisational action selection problem was proposed in Jacob and Magerko (2018) as "creative arc" negotiation during the improvisation as intrinsic motivation for the agent's decision-making, inspired by various aesthetic arcs across several artistic media. A *creative arc* is defined conveniently (but reductively) as a continuous trajectory through a multidimensional *creative space*, currently consisting of *novelty*, *unexpectedness*, and *quality* dimensions. Perceived or generated actions are evaluated computationally and localized to points in the creative space during the performance.

Creative arc negotiation within gestural and object-based movement improvisation was applied to the performance of an improv theater game called *Props*. Improvisers playing the Props game pretend that a given abstract prop is some real-world or fictional object and take turns to use that prop to enact imaginative mimed actions. A virtual reality (VR) installation called the *Robot Improv Circus* was created as a test bed and technical probe to study human-agent improvisation within the Props game domain.

The CARNIVAL architecture (see (Jacob and Magerko 2018) for details) enables improvisational agents to negotiate creative arcs with people. It uses interruptible search over a learned action space in order to choose the closest action to a target point on the creative arc during its turn.

CARNIVAL consists of a parameterizable action generator to perform heuristic search over the action space, improvisational reasoning strategies for guiding search, and creativity evaluation models to localize actions in the creative space.

The parameterizable action generator is a fundamental module used to search the agent's action space for candidate actions that are evaluated by the other parts of the system. Additionally, the generator's representation of the action space constrains the types and implementations of improvisational reasoning strategies for guiding search. Thus the design and evaluation of the generator are vital to the computational creativity problem of human-agent improvisational action selection studied in CARNIVAL.

The action generator for the Robot Improv Circus installation was designed to answer the following research question. ***What representations and processes enable an agent to search a learned object-based interaction space in order to generate believable, recognizable, and high-quality pretend action variants with similar abstract props?*** In order to investigate this question, a deep neural architecture was implemented and evaluated on its capacity for generating believable, recognizable, and high-quality variants of object interactions. This architecture and its variants were trained on mimed human-object actions with props in VR.

A novel feature vector representation of object physical attributes (adapted from (Varadarajan and Vincze 2012)) as an aggregation of part attributes was developed and used to learn a mapping between the physical attributes of objects and a data set of mimed human actions using those objects. Conditioning the mapping this way, enabled the application of learned actions to other physically similar objects. Additionally, the mapping and generator together, form a model of affordance-based action generation since the architecture uses it to constrain generation to actions that are physically suitable or afforded by an object (Norman 1988).

Related Work

Gestural creativity has been modeled in several disciplines, such as choreography synthesis, robotics, and embodied conversational agents. Gesture synthesis systems try to create parameterized, natural, and expressive gestures by following a similar pipeline: input to gesture planner, selection by statistical model, and modification by final component (Ng-Thow-Hing, Luo, and Okita 2010).

Generative choreography systems such as Ikeuchi (2008) and Offi et al. (2012) used segmented music measures as a conditioning input to their generative choreography systems. The most statistically likely candidate dance segments from a pre-authored database were then chosen based on the music inputs and combined to create smooth transitions. Embodied conversational agents create gestures from speech, text, or video clips. Mancini and Castellano (2007) used video tracking and analysis to create an agent capable of mimicking detected expressivity. Kipp et al. (2007) focused on creating natural gestures in virtual agents by using g-units to create continuous flowing movements from gesture segments. Previous models of gestural creativity have been successful in mimicking tasks, but because of the open-ended



Figure 2: A view of the virtual agent miming an action using a prop in the Robot Improv Circus VR installation

action space, a deep generative model was chosen instead of a traditional statistical model.

Gesture synthesis has made significant advances through deep generative models such as Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs). They have proven to be particularly useful for generating novel gestures and choreography with minimal feature engineering by hand. Augello et al. (2017) employed a vanilla VAE trained on a data set of human dance movements to generate robot dance movements. Similar work by Kiasari, Moirangthem, and Lee (2018) focused on combining VAEs and GANs to produce sequences of stylized actions. Their model utilized latent variables from the autoencoder as input to the GAN's discriminator network, while the input to the GAN's generator network was conditioned using action labels and initial poses of the generated action sequences. Our architecture also seeks to control the mode of the generated data through conditioning but adds conditioning both at input and latent space sampling stages since we draw inference directly from the latent space.

Recurrent Neural Networks (RNNs), notably Long Short-Term Memory (LSTM) networks, have been more commonly used for sequential motion generation. Researchers have exploited the hidden Markov model process underlying motion and choreography by using RNN models that combine distributed hidden states and non-linear dynamics. The results are evident in choreographic support (Crnkovic-Friis and Crnkovic-Friis 2016; Tang, Jia, and Mao 2018) and motion synthesis (Holden, Saito, and Komura 2016; Habibie et al. 2017). Our approach extends previous work by conditioning RNN-based generative models for gesture synthesis and preserving local/regional coherence by grouping multiple poses within temporal proximity.

Robot Improv Circus

The Robot Improv Circus is a VR installation for people to play the Props game with a virtual agent. The experience takes place on the stage of a robot circus, where improv is the main event. Participants take turns with the virtual agent to mime pretend actions using abstract props as a real-world or fictional object in imaginative ways in order to create a proto-narrative with the agent.

The VR experience consists of a trial round followed by

a small number of game rounds. Each performer is given a new prop every turn and each round consists of 5-7 turns. The goal of the game is to create a proto-narrative by taking turns miming actions with the prop. Performers hit a buzzer after enacting their actions to signal the end of their turn.

As an example, after receiving a prop shaped like a cube, the VR user might pretend that the prop is a hat and mime putting it on. She then hits the buzzer to end her turn. A new prop, shaped like a long flattened cone, appears in front of the agent who pretends to comb its hair using it as a comb. The agent speaks and displays a speech bubble that reads, "I am combing with a comb" (like in fig. 2). The speech and speech bubbles were added to encourage dialogue and increase recognizability of the mimed actions.

The Robot Improv Circus is exhibited in a circus tent (see fig. ??). The installation has two large displays that act as portals for a real-world audience to view the virtual circus stage. They can watch, applaud, and provide positive feedback to participants in VR, visible as floating emoji above the robot audience in VR.

DeepIMAGINATION

DeepIMAGINATION (Deep IMprovised Action Generation through INteractive Affordance-based exploraTION) is responsible for generating candidate actions for consideration elsewhere in the CARNIVAL architecture. The module represents and reasons about props using their physical attributes and a learned model of object feature vectors allowing learned actions to be generalized to props with similar componential physical attributes that it may not have seen before. The search through the agent's action space is implemented as strategy-guided sampling from the latent space of a conditional variational autoencoder (CVAE) (Sohn, Lee, and Yan 2015) conditioned on the physical attributes of the props making use of the properties of the VAE latent space. The followings sections describe the representations and CVAE architectural variants explored.

Physical Attributes Feature Vector Representation

The physical attributes of a given prop are represented as a fixed-length feature vector. The encoded value is obtained by decomposing the prop into a set of parts that each correspond to a shape primitive with (optional) deformations applied to it. These individual parts are then coded/parsed to obtain a set of binary physical attributes features.

The physical attributes feature set represents the part's shape primitive, size, thickness, flatness, concavity, taper, rigidity, curvature, hole size, and whether a digit/symbol is signified. The feature set was chosen by extending from affordance representation ontologies such as (Varadarajan and Vincze 2012). The physical attributes feature values for each part are then aggregated for the entire prop by summing them together and normalizing them using the maximum count for any feature in the data set. The encoded value represents the normalized counts of each physical attribute feature for the prop across all parts. For example, a barbell-shaped prop might be two flattened spheres connected by a long, thin cylinder. The encoding is currently done by hand given the small number of props and focus of the research.

Gesture Feature Vector Representation

The conditional variational autoencoder (CVAE) models were trained on almost 900 mimed actions of length from 3.3 to 10 seconds collected from five novice improvisers pretending the ambiguously shaped props to be real-world objects (e.g., ladles, golf clubs, and swords) within a VR data collection environment. Each training data point was represented as a single vector with sequential body poses concatenated together and zero-padded as necessary. Models were trained using either a 27000-dimensional or a 16000-dimensional vector representation. The 27000-dimensional vector used 30 features per frame, recorded at 90 FPS for 10 seconds. The 16000-dimensional vector used 35 features per frame at 45 FPS for 10 seconds with 250 entries of zero padding. Each pose consisted of normalized location data (position and rotation) for the user's head and hands in the VR system (and the character's pelvis, calculated using inverse kinematics). The 27000-dimensional representation also had two flags for the VR controllers' grab object button states. The 16000-dimensional representation directly included normalized location data for the prop instead.

Conditional Variational Autoencoder (CVAE) Architecture

The encoder and decoder were both conditioned on the physical attribute vectors of the props used to perform the actions using input concatenation. The encoder reduces the high-dimensional input into a low-dimensional latent space, and the decoder reconstructs a sampled latent vector back into the input space. Fig. 3 depicts how this was done using 1-dimensional convolutional layers and 1-dimensional transposed convolutional layers in the encoder and decoder, respectively. Dropout layers were used for regularization. A recurrent CVAE variant is described in a later subsection.

The network was implemented in TensorFlow and trained with the ADAM optimizer (Kingma and Ba 2014). Given an input distribution X , a latent distribution z and a conditioning distribution c , the CVAE loss function is defined as:

$$L(X, z, c) = E[\log P(X|z, c)] + D_{KL}[Q(z|X, c) || P(z|c)] \quad (1)$$

In other words, the loss function is the sum of the decoder's reconstruction loss and the encoder's Kullback-Leibler divergence loss, both conditioned on the physical attributes distribution. Training the network is made possible by using the re-parameterization trick (Kingma and Welling 2013):

$$z = \mu(X, c) + \epsilon \Sigma^{\frac{1}{2}}(X, c), \text{ where } \epsilon \sim \mathcal{N}(0, 1) \quad (2)$$

During generation, the model's latent space is repeatedly sampled at specific locations provided by the CARNIVAL architecture's improvisational response strategies, based on the current improvisational context occurring. The DeepIMAGINATION module generates candidate actions conditioned on the physical attributes of the given prop. Candidate actions are evaluated by the creativity evaluation models of the CARNIVAL architecture, and the candidate action that is closest to the next target point on the agent's creative

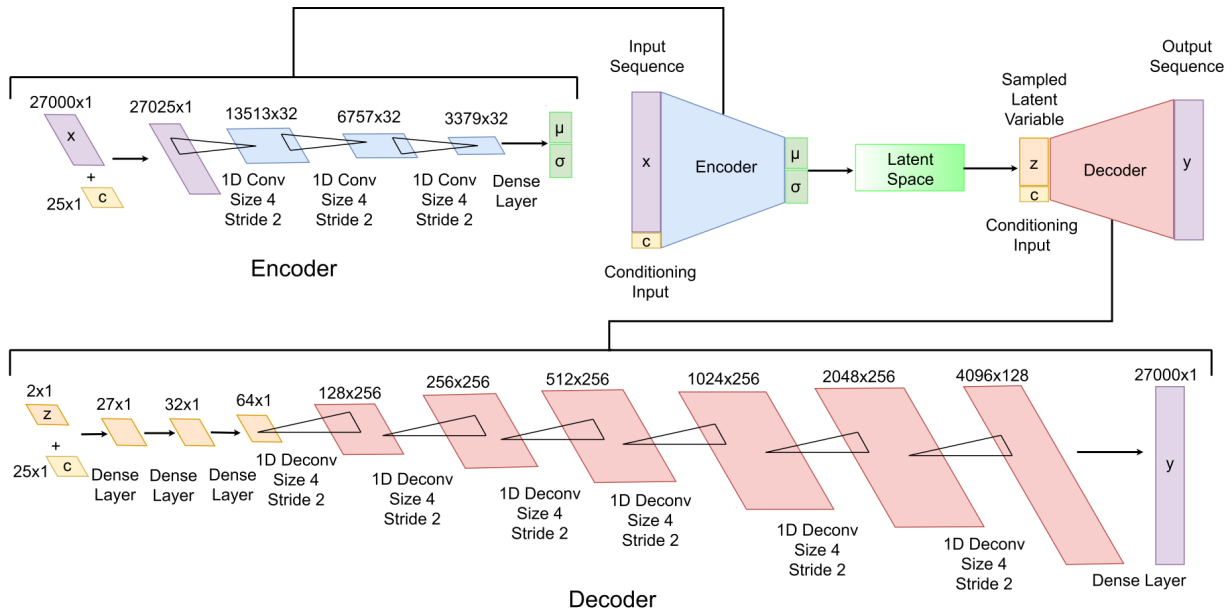


Figure 3: A convolutional variant of the DeepIMAGINATION architecture with $(27000, 1)$ shaped input gesture and 2D latent space. General CVAE architecture shown in upper right quadrant. Zoomed-in views of encoder and decoder in upper left and bottom respectively. Dropout layers not shown but applied between every 1D convolution and transposed convolution layer.

arc for its current turn is selected. The generated output is then used within the Robot Improv Circus VR experience to control the rigged character model of a robot avatar using inverse kinematics (to control the character's other joints). Exponential moving average smoothing of joint trajectories is applied due to the IK-induced shakiness of the actions.

A total of eight architecture variants were designed and trained, including four convolutional models and four recurrent models. The variants were implemented for performance evaluation and selection. The convolutional architectures only differed in their input vector representations. The four recurrent models used either a standard RNN architecture or an architecture based on the MusicVAE network (Roberts et al. 2018). The two groups of RNN variants were also trained on different input vector representations.

Convolutional Variants It is helpful to think of the different input vector representations $((27000, 1), (16000, 1), (900, 30), \text{ and } (450, 35))$ for convolutional models in terms of the number of channels in the input data. The data was first represented with one channel, that is, 27000 and 16000 dimensional vectors were reshaped to $(27000, 1)$ and $(16000, 1)$ dimensional tensors, respectively. In another representation, the number of channels corresponded to the number of features per body pose frame - i.e., 27000 dimensional vectors were reshaped to $(900, 30)$ tensors while the 16000 dimensional vectors were reshaped to $(450, 35)$ tensors (disregarding the zero padding).

Recurrent Variants The Recurrent Neural Network versions of CVAE were implemented using long short-term (LSTM) layers. Both the encoders and decoders of the Vanilla RNN implementation include single layers of bidirectional LSTMs that represented information for each

frame concatenated with the physical attributes vector. Based on results from Roberts et al., where vanilla RNN-based decoders sometimes had poor sampling and reconstruction performance, a hierarchical RNN architecture for the decoder was designed based on their MusicVAE architecture. In this variant, the latent vector z is first passed through a fully connected layer to initialize the state of the Conductor layer, which is composed of a unidirectional LSTM layer. The output of the conductor layer is then passed as initialization for the bottom LSTM layers, where each frame vector from Conductor layer, concatenated with the output of previous bottom layer LSTM, is used as initialization for the bottom layer LSTM of next time interval. The outputs of each bottom layer LSTM are then concatenated and flattened to match the input tensor shape.

Methodology

The DeepIMAGINATION module is the parameterizable action generator for searching the agent's action space in the CARNIVAL architecture. It was designed to investigate, "what representations and processes enable an agent to search a learned object-based interaction space in order to generate believable, recognizable, and high-quality pretend action variants with similar abstract props?" Therefore, our evaluation plan for the DeepIMAGINATION module consisted of the following questions.

- EQ1 Which CVAE variant best learned the distribution of human-object interactions?
- EQ2 Does the architecture allow an agent to generate action variants that are believable, recognizable, and high-quality compared to human actions?

Benchmarking Error

Standard quantitative evaluation metrics were chosen to benchmark the eight architecture variants and evaluate EQ1. These metrics were Euclidean distance, root mean squared error, and cosine similarity, as well as the final epoch mean loss. Each architecture was evaluated using a model trained over 40000 iterations (approximately 2850 epochs with a batch size of 64) using a 10% validation split. The trained model was then fed the entire data set, calculating metrics between each input vector and the reconstructed vector created by the CVAE. The mean, median, and standard deviation of each metric along with training and validation loss from the final epoch of training are reported in the next subsection (see Table 1). For qualitative comparisons, visual inspection within our Unity3D environment was used to compare the different generated actions.

Mechanical Turk Study

We created multiple surveys using Amazon's Mechanical Turk platform that assessed the believability, quality, and recognizability of four data sets related to actions from DeepIMAGINATION. The experiment was conducted to address the evaluation question EQ2 described above. Each of the four data sets consisted of 40 gestures performed by a robot character in VR across 20 props from the Robot Improv Circus. A GIF was recorded of the robot character performing two actions with each prop for a total of 160 actions across all four datasets. These GIFs were then evaluated by remote workers on the Mechanical Turk platform.

Data Sets The human-generated data set comprised actions performed by a human in VR with a robot avatar. This set of human gestures was then passed through DeepIMAGINATION in various conditions to generate three additional data sets of actions with the same robot avatar. The direct output of the autoencoding made up the agent mimicry data set as it represented the agent's interpretation of human gestures. The third and fourth data sets were made up of near and far action variants (respectively) of the agent mimicry data set. They were generated by sampling points that were nearby and further away (respectively) from the mimicry gestures in the CVAE model's latent space. The same robot avatar performed these actions as well.

Each survey required the participant to watch either one or two recorded GIFs of actions (depending on the task) from one of the four datasets and answer a few questions about that GIF. In each survey, the human data set made up the human actions, and the other three data sets made up the computer generated actions. There were 80 participants for tasks with single GIFs and 60 participants for tasks with two GIF comparisons. Each participant worked on 20 GIFs out of the entire data set of GIFs.

Believability In order to assess the believability of the actions, two survey tasks were given to Mechanical Turk workers. In the first survey, each participant watched a single GIF at a time and answered if they believed the action was performed by a human in VR or generated by a computer program. The comparison was made in order to

evaluate whether participants could tell the difference between computer-generated (CG) actions and human actions between each data set. The hypothesis was that differences would be seen between the discrimination accuracy of generated actions according to which of the three CG data sets was being evaluated (indicating that at least some groups of CG actions were as believable as human actions).

A second study was run that asked people to compare a human action from the human actions data set with a CG action from one of the other three datasets and asked the participant to identify which action they believed was generated by a computer, if both were generated by a computer, or neither was generated by a computer. The test helped to clarify whether participants thought that computer-generated actions were human actions when directly comparing the two. The test also indicated how believable the CG GIFs were. If the participants had low accuracy in determining the identity of the computer-generated GIF, it would indicate that the computer-generated GIFs were believable. The hypothesis was that there would be significant differences in recognition accuracy across groups, indicating that the CG actions were mistaken for human actions in some of the groups.

Recognizability The recognizability of the actions in our data sets was assessed in terms of how identifiable the pretend object and pretend action were that the character in the GIF was portraying. The survey asked participants to select what they believed the robot character was most likely enacting from a list of three options. The options were similar to stabbing with a sword or eating with a spoon. High accuracy in identifying the actions and objects shown in the GIF would indicate that the portrayal was recognizable overall. Our hypothesis was that comparable recognition accuracy across groups would be seen showing that the CG action sets were equally recognizable to human actions.

Quality Participants were asked to determine the quality of the GIFs through two tasks. In the first one, participants were asked to rate the smoothness and quality on a 5-point Likert scale by looking at a GIF and evaluating it on its own. They were also asked to state what they thought were criteria for quality in this domain before rating any GIFs and were asked to use those criteria strictly during rating.

The second task was a forced choice condition. Participants were asked which action they thought was smoother and of higher quality. Each participant was asked to define quality at the beginning of the survey and use those criteria strictly while rating the GIFs for quality.

The two measures (smoothness and user-defined quality) were used together to assess the overall quality of each action in both tasks. If smoothness and user-defined quality were high for each action, it would indicate that the overall quality was high. Our hypothesis was that there would be comparable quality and smoothness ratings across groups.

Results

Benchmarking Error

The results of the standard evaluation metrics that were used on the eight architecture variants are shown in Table 1 and

the final epoch mean losses are shown in Table 2. Comparing the Conv. (16K, 1) architecture with the Conv. (27K, 1) architecture, the former performs better in all metrics except for training loss. However, Conv. (16K, 1) model's validation loss is lower, which provides stronger evidence that the (16000, 1) representation allows for a better reconstruction. Both of the reshaped convolutional feature vector representations outperformed their respective un-reshaped versions. The result shows more support that the reshaping helped the CVAE learn more beneficial relationships when the per pose features were split up across channels.

The RNN-based models seem to generalize better than the convolutional variants, as the Vanilla 16K model performs the best in validation loss (shown in Table 2). In contrast to RNN variants, convolutions based models overfit drastically on the training set. The variance in losses between 27K and 16K RNN based models in Table 1 and 2 shows that RNN-based models are resilient to reductions in input dimensions.

Mechanical Turk Study

Believability The task of detecting whether a given GIF was human performed or CG was treated as a binary classification task between the performance of the participants on the human data set in comparison to their performance on each of the other three data sets. The lower the participant accuracy, the stronger would be the evidence that the CG actions were believable. In order to analyze the participant responses, a confusion matrix was created for the four sets of comparisons: human vs. all CG, human vs. agent mimicry, human vs. near variant, and human vs. far variant. The F1 scores for the four conditions were: 0.5251, 0.7154, 0.7163, and 0.671. Additionally, the Matthews Correlation Coefficients for the four conditions were: 0.3308, 0.4237, 0.426, and 0.2912, respectively (all weak positive correlations).

The results above showed that the believability of the CG actions was comparable to that of the human actions in the single GIF rating task when human vs. all CG or human vs. far variant conditions were considered. The fact that far variants scored the highest in comparison to agent mimicry and near variants was surprising since it was the least close to the corresponding human point in the latent space. However, it is possible that it was close to some other human point and thus ended up generating believable actions.

The claim that human-performed and CG GIFs could be confused for each other was further bolstered by treating the participants as raters and calculating an inter-rater reliability (IRR) score for how they rated whether the GIFs were human-performed or CG. The IRR score, Krippendorff's alpha, across all data types, was calculated to be 0.23925, showing only a slight agreement between participants about the origin of the action. The result was interpreted to mean that the human-performed and CG GIFs could not reliably be determined across participants and data sets.

Responses from the forced choice believability evaluation task between two action GIFs was assessed by treating it as a multi-class classification problem. The options given to participants were – CG action on the left, CG action on the right, both CG actions, and neither CG actions. As a

reminder, poor participant performance on this task would be indicative that the CG actions were highly believable.

A four-class confusion matrix was created for the four responses possible, once each for human vs. agent mimicry, human vs. near variant, and human vs. far variant. In that order, the F1 scores were 0.8157, 0.7925, and 0.7678, respectively. The Matthews Correlation Coefficient was calculated respectively, to be 0.5414, 0.5938, and 0.4931 (strong positive correlations). Both results were calculated using micro-averaging due to the multi-class condition. The result indicated that when compared directly side-by-side to a human-performed action, participants were able to identify the human-performed action with relatively high accuracy, indicating that the actions were not as believable as desirable when compared directly against a human-performed action.

Recognizability Participants were asked to identify the actions performed by robot characters when assessing the recognizability of actions. Their mean accuracy (standard deviation in parenthesis) was determined across the different data sets ordered as human, agent mimicry, near variant, and far variant as 0.64 (0.26), 0.37 (0.24), 0.41 (0.23), and 0.33 (0.27). The median accuracy values for the same groups were 0.66, 0.4, 0.4, and 0.30. This outcome is a negative result that shows that recognizability for CG actions was comparable to random guessing, while human-performed actions were twice as likely to be recognized correctly.

A Shapiro-Wilk test found a non-normal distribution for the accuracy data. Therefore, a Kruskal-Wallis omnibus rank sum test was computed on the data. The results were found to be significant, and the null hypothesis was rejected with a confidence level = $5.505771 \cdot 10^{-17}$. A Dunns test adjusted with Benjamini-Hochberg FDR showed that all the negative result relationships between the human data and the CG data were significant (all confidence levels < 0.019641).

Quality When assessing forced choice smoothness and quality of each action, the medians were calculated for the Likert scale responses and chi-squared tests were calculated for the human data compared to each of the three data types to see if there were significant associations between the types of data and the Likert scale responses for smoothness (or high quality respectively). For single GIF smoothness, the median values for human, agent mimicry, near variants, and far variants were 4, 2, 2, 3 on a 1 - 5 scale from not at all smooth to very smooth. The chi-squared test reported significance with $\tilde{\chi}^2 = 304.9299$ and a confidence level of < 0.00001 . For single GIF user-defined quality, the median scores reported for the same data sets were 4, 3, 3, 3 on a similar scale from very poor quality to very high quality. The chi-squared test reported significance with $\tilde{\chi}^2 = 265.4731$ and a confidence level of < 0.00001 .

When assessing forced choice smoothness and quality of each action, the percentage of results that were considered smoother (or higher-quality respectively) was recorded along with chi-squared tests that were calculated for human data compared to each of the three data types. The test was conducted to see if there were significant associations between the types of data and the selection of the human or computer action as more smooth (or high qual-

Table 1: Evaluation metrics. Note: (+) means higher is better and (-) means lower is better.

Architecture	Euclidean Distance (-)			Root Mean Squared Error (-)			Cosine Similarity (+)		
	Mean	Median	Std Dev	Mean	Median	Std Dev	Mean	Median	Std Dev
Conv. (27K, 1)	11.160	9.628	4.933	0.068	0.059	0.030	0.972	0.981	0.002
Conv. (16K, 1)	3.404	2.640	2.523	0.027	0.021	0.020	0.997	0.999	0.005
Conv. (900, 30)	2.975	1.871	3.125	0.019	0.011	0.019	0.996	0.999	0.007
Conv. (450, 35)	2.763	1.656	3.033	0.022	0.013	0.024	0.997	1.000	0.007
Vanilla 16K	6.634	6.473	1.278	0.052	0.051	0.010	0.993	0.993	0.003
Conductor 16K	6.114	5.976	1.294	0.048	0.047	0.010	0.994	0.994	0.002
Vanilla 27K	6.473	5.830	2.491	0.039	0.035	0.015	0.991	0.993	0.006
Conductor 27K	6.469	5.811	2.433	0.039	0.035	0.015	0.991	0.993	0.006

Table 2: Final epoch mean loss (lower is better).

Architecture	Training Loss	Validation Loss
Conv. (27K, 1)	9.83	145.28
Conv. (16K, 1)	14.65	119.00
Conv. (900, 30)	20.88	139.81
Conv. (450, 35)	21.13	138.49
Vanilla 16K	34.81	38.01
Conductor 16K	42.39	51.09
Vanilla 27K	39.86	48.56
Conductor 27K	47.44	56.17

ity respectively). For smoothness, human data was chosen as smoother 75.63% against agent mimicry, 77.54% against near variants, 75.30% against far variants, and 76.14% overall against all CG actions. There were no significant differences found between the groups, with $\tilde{\chi}^2 = 0.6701$ at a confidence level of < 0.05 . For user-defined quality, the percentage of responses where human data was chosen as higher-quality was 73.58%, 78.26%, 76.74%, and 76.14% for the same ordering as smoothness. There was no significant association found either, with $\tilde{\chi}^2 = 2.6957$ at a confidence level of < 0.05 .

Discussion

The action generation module described in this article is a vital part of the CARNIVAL architecture that enables improvisational embodied agents to improvise with people. Thus the evaluation was conducted based on its capabilities as a generator that could create believable, recognizable, and high-quality outputs. However, the larger evaluation design for an architecture that models creativity in such an open-ended and ill-defined domain has been challenging.

The task of evaluating generator outputs out of context could have been unusual for many human evaluators (though perhaps less so for those familiar with prop-based improv theatre). Therefore, the results of the human evaluation task may not truly reflect the agent’s performance within the context of the entire CARNIVAL architecture. Additionally, the benchmarking experiments did evaluate how well the model learned the distribution of human actions, but the perceiv-

able difference between models also needs to be evaluated. As a result, further studies will culminate in observational and in-person evaluation of the entire CARNIVAL architecture as an improvisational partner.

Our CVAE models all significantly overfit the data set due to the small size of data set used for training. Regularization only partially mitigated overfitting. We are currently conducting data collection and doing annotation on collected data to increase the amount of training data available.

A redesigned representation of physical attributes considering prop part ordering and spatial relationships is planned. The added nuance was not an initial priority. The suitability of the representation and its capacity for supporting transfer of learned actions to other props with similar physical attributes will also be evaluated.

We are planning to experiment with adversarial training of our architecture variants. The experiment would solve some of the challenges with the CVAE generation though it does introduce other difficulties like increased modal collapse that will need to be addressed. Additionally, adversarial training is challenging to perform at the moment, since we have minimal data. Another potential solution around the lack of data could be self-supervision on the unlabeled examples that have been collected but not annotated.

Additionally, even though the RNN based models show exceptional performances in terms of robustness and generalization, the loss values for the models are still relatively high compared to some of the best convolutional models in terms of training loss, mean Euclidean distance, and root mean squared loss. One potential improvement that can be made for the RNN based architectures is to increase the hidden layer size as well as stacking multiple RNN layers together to increase potential expressiveness of the models.

Conclusion

This article described a deep learning approach to generating candidate actions within the CARNIVAL architecture in order to address the improvisational action selection problem within human-agent embodied improvisation with objects. Four convolutional variants and four RNN variants of the proposed CVAE model were used to generate agent movements based on human inputs and prop physical attributes. In addition, the performance of those models was

analyzed using benchmarking metrics as well as Mechanical Turk studies to evaluate their believability, recognizability, and quality among observers.

The results showed that our models could successfully learn the distribution of training data. In terms of a subjective evaluation from human subjects on Mechanical Turk, the generated results were relatively believable as long as they were not simultaneously compared against human actions. They did not seem very recognizable, however. Therefore, we have implemented an agent speech bubble in the Robot Improv Circus that serves as a way to increase recognizability and communicate about agent intent. Finally, while human actions were consistently rated smoother and higher quality than CG actions, the ratings themselves, especially user-defined quality, were positive for all CG actions.

Acknowledgements

The authors would like to acknowledge the other researchers who supported this project as well as the reviewers who provided valuable feedback. This project is funded by a Creative Curricular Initiatives grant from the Georgia Tech Office of the Arts and Georgia Tech Arts Council.

References

- Augello, A.; Cipolla, E.; Infantino, I.; Manfré, A.; Pilato, G.; and Vella, F. 2017. Creative robot dance with variational encoder. *CoRR* abs/1707.01489.
- Crnkovic-Friis, L., and Crnkovic-Friis, L. 2016. Generative choreography using deep learning. *CoRR* abs/1605.06921.
- Davis, N.; Hsiao, C.-P.; Yashraj Singh, K.; Li, L.; and Magerko, B. 2016. Empirically studying participatory sense-making in abstract drawing with a co-creative cognitive agent. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, 196–207. ACM.
- Habibie, I.; Holden, D.; Schwarz, J.; Yearsley, J.; and Komura, T. 2017. A recurrent variational autoencoder for human motion synthesis. In *BMVC*.
- Hoffman, G., and Weinberg, G. 2010. Gesture-based human-robot jazz improvisation. In *Proceedings - IEEE International Conference on Robotics and Automation*, 582–587.
- Holden, D.; Saito, J.; and Komura, T. 2016. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.* 35(4):138:1–138:11.
- Ikeuchi, T. S. K. 2008. Synthesis of dance performance based on analyses of human motion and music.
- Jacob, M., and Magerko, B. 2015. Interaction-based Authoring for Scalable Co-creative Agents. In *Proceedings of the Sixth International Conference on Computational Creativity (ICCC 2015)*.
- Jacob, M., and Magerko, B. 2018. Creative arcs in improvised human-computer embodied performances. In *Proceedings of the 13th International Conference on the Foundations of Digital Games*, 62. ACM.
- Kiasari, M. A.; Moirangthem, D. S.; and Lee, M. 2018. Human action generation with generative adversarial networks. *CoRR* abs/1805.10416.
- Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980.
- Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.
- Kipp, M.; Neff, M.; Kipp, K. H.; and Albrecht, I. 2007. Towards natural gesture synthesis: Evaluating gesture units in a data-driven approach to gesture synthesis. In *IVA*.
- Magerko, B.; Permar, J.; Jacob, M.; Comerford, M.; and Smith, J. 2014. An Overview of Computational Co-creative Pretend Play with a Human. In *Proceedings of First Workshop on Playful Virtual Characters at the Fourteenth Annual Conference on Intelligent Virtual Agents*.
- Mancini, M., and Castellano, G. 2007. Real-time analysis and synthesis of emotional gesture expressivity.
- Mathewson, K. W., and Mirowski, P. 2017. Improvised theatre alongside artificial intelligences. In *Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference*.
- Ng-Thow-Hing, V.; Luo, P.; and Okita, S. Y. 2010. Synchronized gesture and speech production for humanoid robots. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems* 4617–4624.
- Norman, D. A. 1988. The psychology of everyday things.
- Ofli, F.; Erzin, E.; Yemez, Y.; and Tekalp, A. M. 2012. Learn2dance: Learning statistical music-to-dance mappings for choreography synthesis. *IEEE Transactions on Multimedia* 14:747–759.
- O’Neill, B.; Piplica, A.; Fuller, D.; and Magerko, B. 2011. A knowledge-based framework for the collaborative improvisation of scene introductions. In *Proceedings of the 4th International Conference on Interactive Digital Storytelling*, volume 7069 LNCS, 85–96.
- Reidsma, D.; van Welbergen, H.; Poppe, R.; Bos, P.; and Nijholt, A. 2006. Towards Bi-directional Dancing Interaction. In *Entertainment Computing - ICEC 2006*, 1–12.
- Roberts, A.; Engel, J.; Raffel, C.; Hawthorne, C.; and Eck, D. 2018. A hierarchical latent vector model for learning long-term structure in music. In *ICML*.
- Sohn, K.; Lee, H.; and Yan, X. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, 3483–3491.
- Spierling, U., and Szilas, N. 2009. Authoring issues beyond tools. In *Joint International Conference on Interactive Digital Storytelling*, 50–61. Springer.
- Tang, T.; Jia, J.; and Mao, H. 2018. Dance with melody: An lstm-autoencoder approach to music-oriented dance synthesis. In *Proceedings of the 26th ACM International Conference on Multimedia*, MM ’18, 1598–1606. New York, NY, USA: ACM.
- Varadarajan, K. M., and Vincze, M. 2012. Afnet: The affordance network. In *Asian Conference on Computer Vision*, 512–523. Springer.

Exploring Conditioning for Generative Music Systems with Human-Interpretable Controls

Nicholas Meade^{*,1}, Nicholas Barreyre^{*,1}, Scott C. Lowe^{1,2}, Sageev Oore^{1,2}

¹Faculty of Computer Science, Dalhousie University, Halifax, NS, Canada

²Vector Institute, Toronto, ON, Canada

nicholas.meade@dal.ca, nbarreyre@dal.ca, scottclowe@gmail.com, sageev@dal.ca

Abstract

Performance RNN is a machine-learning system designed primarily for the generation of solo piano performances using an event-based (rather than audio) representation. More specifically, Performance RNN is a long short-term memory (LSTM) based recurrent neural network that models polyphonic music with expressive timing and dynamics (Oore et al., 2018). The neural network uses a simple language model based on the Musical Instrument Digital Interface (MIDI) file format. Performance RNN is trained on the *e-Piano Junior Competition* Dataset (International Piano e-Competition, 2018), a collection of solo piano performances by expert pianists. As an artistic tool, one of the limitations of the original model has been the lack of useable controls. The standard form of Performance RNN can generate interesting pieces, but little control is provided over *what* specifically is generated. This paper explores a set of conditioning-based controls used to influence the generation process.

Introduction

Computational, automated, and stochastic generation of music are pursuits of long-standing interest (Hedges, 1978); recently there has been an increasing body of research interest in these subfields (Huang et al., 2019a; Dieleman, van den Oord, and Simonyan; Herremans, Chuan, and Chew, 2017; Huang et al., 2019b; Payne, 2019; Roberts et al., 2018).

In a typical auto-regressive language model, the system generates a discrete probability distribution $P(\text{event}_0)$, samples from that distribution, and then uses its own sampled event history to condition the probability distribution over the next event to be predicted. An RNN model with a finite vocabulary is continually predicting $P(\text{event}_t | \text{event}_{i < t})$, where each event_t is drawn from the vocabulary. PerformanceRNN, for example, is such a language model applied in a musical setting to generate expressive piano improvisations (Oore et al., 2018).

One can adapt an auto-regressive language model so that its predictions are conditioned not only on the past events, but also on an externally-specified signal. For example, Malik and Ek (2017) condition expressive generation on a score, and Donahue, Simon, and Dieleman (2018) incorporate melodic pitch contours to provide very nice control over the generative mechanism.

^{*}Equal contribution; ordering determined by coin toss.

In this work we explore further ways to provide the user with control over Performance RNN’s generated musical output through a variety of conditioning signals, considered both individually and jointly. We begin by describing the data set used to train the system.

The architecture we use to pass in control signals to Performance RNN is shown in Figure 1.

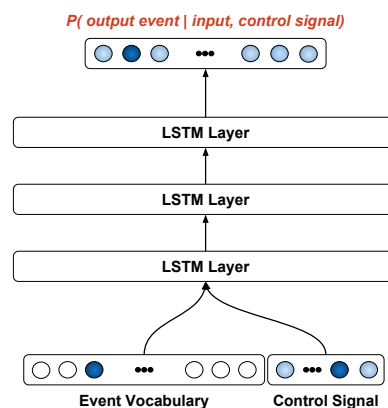


Figure 1: Performance RNN Architecture with Control Signals

Dataset

Similar to work by Oore et al. (2018), we used the *e-Piano Junior Competition* Dataset (International Piano e-Competition, 2018) as training data for our models. The raw MIDI files were converted to the Performance RNN representation, with a 388 word vocabulary consisting of 128 note-on, 128 note-off, 100 time-shift, and 32 velocity events. Performances are modelled as sequences of these events, and are fed into the neural network using a one-hot vector encoding at each step.¹

¹The code both for the representation conversion and for the basic model were derived from the publicly available Magenta repository at <https://github.com/tensorflow/magenta/>.

This dataset consists of 2750 performances by skilled pianists². The performances were split into training and validation partitions (90:10). Additionally, we augmented the dataset with transpositions of up and down all intervals up to five or six semitones (spanning a full octave), and with temporal stretches/compression factors of 2.5% and 5%, increasing the number of training samples 35-fold.

We train the model on 30 s segments from the MIDI performances using teacher forcing.

To achieve a conditional variant of Performance RNN, an additional feature vector is provided to the model along with each event, which we refer to as a *control signal*. At training time, the control signal provides additional information about the performance using metadata such as the composer of the piece. When generating samples from the model, we can then constrain it to output a performance in the style of a single composer, for instance.

Various control signals were used, each described in more detail in the following sections. Briefly, we had 3 sources for our control signals.

- Signals corresponding to the local statistics of the clip within the performance (note density, note velocity, and relative positioning within the piece).
- Metadata directly available from the dataset (composer, and from this, attributes of the composer such as their year of birth).
- Metadata extracted from the titles of the pieces (key, tempo and form).

For the first two sources, the metadata had complete coverage across the dataset. However, only a minority of the titles indicated the key, tempo, or form of the piece.

Control Signals

In this section, we detail the control signals we explored, and the results for each. The generated results mentioned below, along with additional samples, are all available at doi:10.5281/zenodo.2883725

Composer-based Conditioning

Though there are commonalities between composers, each composer has their own style of composition. Being able to condition the model to generate music in the style of a particular composer would give us a useful control mechanism.

There are 114 different composers in the dataset, though some occur more frequently than other, less popular ones. The distribution is approximately log-normal; around 53% of the performances were pieces with one of the five most popular composers — Chopin, Beethoven, Liszt, Schubert, and Bach. From the model’s perspective, most sequences in its world are composed by one of these five men, and an unconditioned model will hence generate music in one of their styles more often than not. This highlights the utility of being able to select the composer whose style should be emulated.

²More competition data has been released, allowing us to increase the size of the dataset from 1400 to 2750 performances.

We explored two types of control signals using the composer, either conditioning on the composer of the piece directly, or clustering the composers into groups and conditioning on the group.

Individual Composers We used a 114-length feature vector representing the fraction of each composers’ contribution to the training example. Usually, this was a one-hot encoding; however, certain performances in the dataset were composed by multiple individuals. These cases may arise, for instance, when one composer has written a piece for harpsichord and another composer has transcribed it into a piece for piano; in such cases all composers have influenced the final piece. We then assigned equal weighting in the control signal of $1/n_c$, where n_c is the number of composers.

We found that the model was able to express some of the stylistic differences between the composers (see Additional Materials to hear samples). However, the distribution of the number of pieces per composer in the dataset has a long tail, and there were many composers for which only a single performance was available (around 29% of composers only had a single piece in the dataset, meaning 1.2% of the performances possess a unique composer). For these one-off composers, it was not possible to both train a model conditioned on them and also to validate the model under that conditioning. Furthermore, such a model could have tendency to overfit on composers associated with very few performances. We comment further on the significance of overfitting (or its insignificance) in the context of an artistic tool in the discussion section.

To quantify how well our composer-conditioned model captured the stylistic differences between composers, we surveyed five professional musicians with expertise in classical piano. We selected five composers — Bach, Beethoven, Chopin, Debussy, and Mozart — and generated eight samples, each 20 s in duration, for each composer. Each participant was given 10 samples (two from each composer) and tasked with rating the stylistic similarity of each clip to each of the five composers. Scores were given from 1–5, where 1 denoted highly dissimilar and 5 was highly similar. On average, participants scored the correct composer with a rating of 2.76 ± 0.18 (SEM) and incorrect composers with a rating of 1.95 ± 0.07 (SEM).

Clustered Composers An expert pianist grouped 46 of the 114 composers into eight clusters (see Table 1) based on style. We used an input analogous to the individual composer control signal, except a 9 bit encoding was used to represent the distribution over the composer clusters instead.

Time-period Conditioning As with any art form, styles of composition evolve over time; many styles of classical music are associated with the period of history in which they originated and proliferated.

Unfortunately, the year of composition of each piece is not part of the metadata for this dataset. As a proxy for the time period in which a piece was written, we used the year of birth of each composer.

We then grouped each performance by the century in which its composer was born: 1600, 1700, 1800, 1900, and

Cluster	Composers
Cluster 1	Balakirev, Bartholdy, Bizet, Brahms, Busoni, Chopin, Grieg, Horowitz, Liszt, Mendelssohn, Moszkowski, Paganini, Saint-Saens, Schubert, Schumann, Strauss, Tchaikovsky, Wagner
Cluster 2	Beethoven
Cluster 3	Bach, Handel, Purcell
Cluster 4	Barber, Bartok, Hindemith, Ligeti, Messiaen, Mussorgsky, Myaskovsky, Prokofiev, Schnittke, Schonberg, Shostakovich, Stravinsky
Cluster 5	Debussy, Ravel
Cluster 6	Clementi, Haydn, Mozart, Pachelbel, Scarlatti
Cluster 7	Rachmaninoff, Scriabin
Cluster 8	Gershwin, Kapustin
Cluster 9	<i>Everyone else</i> (unclustered)

Table 1: Composer clustering, constructed by hand by an expert pianist. From the composers, 46 were clustered into eight groups. The 68 unclustered composers were placed together in an additional, ninth group.

2000 CE. This was similar to the previously mentioned composer clustering, except that the binning of the composers was done based solely on their chronology without considering other factors.

In another variation, we normalized the year of birth of the composer to be a scalar in $[0, 1]$ over the entire dataset, allowing us to interpolate and generate music conditioned on any year between roughly 1650 to 2000 CE.

Composer Latitude, Longitude, and Birth Year Compositional styles are associated not only with a historical period, but also geographical regions. The flow of musical knowledge and influence through both time and space motivated us to use a control signal based on the geographical locale of the composer, in addition to the time period in which it was written.

The city that was most associated with each composer provided us with geographic information about the music. As above, we used the year of birth of the composer as a proxy for the date of authorship of each piece. We used min-max normalization on the latitude, longitude, and year of birth, and represented this control signal as a vector of length three. Thus, one component controlled movement from North to South, another from East to West, and a third, the year, spanned from 1653 to 1972 CE.

For example, in Sample 1 we can hear a sample generated conditionally on a city in Germany in 1685. Intuitively, we expect this to sound somewhat Bach-like. In Sample 2 and Sample 3 we can hear two samples generated conditionally on Warsaw in 1810. We would expect this to somewhat resemble Chopin. Overall, however, the year of birth tended to work more as expected than the location; it is likely that

there was insufficient geographic variety in the data set, so that, for example, french impressionism did not sound particularly recognizable as such. However, extrapolating to regions outside of the training distribution did occasionally produce quite interesting results, even if they did not make “sense” from the point of view of music history.

Discussion Of the four mechanisms for conditioning based on the composer, we found that individual composer conditioning produced the most pleasing samples. One reason why we think that composer clustering did not produce as high quality samples is that only 68 out of the 114 composers were placed into meaningful clusters, while the remaining ones were essentially counted as “unknown”. As we discuss below, there is a potential problem with this.

Title Keyword Conditioning

Along with the MIDI files included as part of the e-Piano Competition data set, we also scraped the title of each performance from the web.

For some pieces, the title provided useful and interpretable information. An example of such a title is *Sonata in D Major, K. 576 (Complete) I. Allegro*, which we can easily determine is in the key D Major, and marked tempo Allegro. The signals that we extracted from the titles were key, tempo, and form of the piece.

Encoding mechanism for partially labelled control signals

As previously noted, some conditioning signals — in particular, signals based on information present in the title — were only available for a subset of the data. For control signals which are always available to condition the model on during training, we supply the corresponding feature vector. However, for control signals which are only partially observable, we must choose some default input to supply to the model in lieu of a feature vector.

Let us denote a multivariate control signal as $c = [c_1, c_2, \dots]$. When the control signal is unavailable, one option is to follow the methodology of PerformanceRNN’s optional conditioning inputs. With this method, when the control signal is unknown it is filled with zeros, and the conditioning signal is prepended by an additional bit, c_0 , which indicates whether the control signal is provided. For instance, the new conditioning input would be $[0, c_1, c_2, \dots]$ when the control is available and $[1, 0, 0, \dots]$ when it is not. We experienced some difficulties using this encoding paradigm, particularly when control signals were very sparsely available — for instance the tempo, which is only provided in the title of around 19% of the pieces. In these cases, the model would generate appropriate outputs only when the bit indicating the absence of the control signal was set to 1.

We can provide some intuition for this failure mode as follows. Let us consider the most extreme case: a control signal which is always absent from the dataset, so $c_0 = 1$ for all samples. In this case, the conditioning signal is always set to $[1, 0, 0, \dots]$; aside from the first bit (corresponding to absence of a control signal) the weights connected to the conditioning signal are all irrelevant to the model and will remain at their initialized values (unless weight-decay is present, in which case they decay to 0). Meanwhile, the

weights connected to the first bit, $w_{c_0}^{(j)}$, are each redundant with the respective neuron’s bias term, $b^{(j)}$. It is now important to the model only that the sum of these two parameters, $\hat{b}^{(j)} = b^{(j)} + w_{c_0}^{(j)}$, is optimized, and the bias term $b^{(j)}$ itself may be very different to the bias term which would be learnt when training a model without the conditioning signal. Without weight-decay, the difference between the two components will be the same after training as it was at initialization, since the parameters receive identical weight updates. After training this model, inputs with $c_0 = 0$ will (clearly) not behave well since as the $w_{c_0}^{(j)}$ terms are necessary to counteract the bias terms $b^{(j)}$ such that the effective bias, $\hat{b}^{(j)}$, is as optimised. Ignoring the rest of the conditioning signal, if we were to include some inputs where $c_0 = 0$ during training, we would expect them to break the symmetry between $b^{(j)}$ and $w_{c_0}^{(j)}$, and eventually $w_{c_0}^{(j)} \rightarrow 0$. However, the magnitude of the weights $b^{(j)}$ and $w_{c_0}^{(j)}$ is of a similar order of magnitude to the activation of the neuron. For a substantial fraction of the neurons where $w_{c_0}^{(j)} > 0$, the neurons will have a negative preactivation whenever $c_0 = 0$, rendering them entirely inactive under a ReLU activation function. Hence if the fraction of training samples where $c_0 = 0$ is insufficiently high, they will fail to break the symmetry before the model finds a local optima, at which many of the neurons are permanently dead for all samples with $c_0 = 0$.

We found better results could be achieved simply by omitting the c_0 bit from the conditioning vector during training. This model can be conceptualised as learning a boosting procedure. A “baseline model” is learnt which is used when the control signal is unknown and the conditioning signal is set to $c_i = 0 \forall i > 0$. But when the conditioning signal is non-zero, the weights $w_{c_i}^{(j)}$ are used to make fine-tuning improvements to the baseline model by increasing or decreasing the activation of each neuron in the first LSTM layer. As a consequence, the residual error of the baseline model is reduced when the control signal is available.

For one-hot control signals, we also found good results by using a uniform distribution across c_i when the true label was unknown (again, omitting a c_0 term).

Major-Minor Conditioning The key of an excerpt of music is informative with regards to the pitches one would expect to dominate within the music, both in terms of number of occurrences and emphasis. By extracting the key signatures that were present in performance titles, we were able to provide a control signal to the model corresponding to the key of the piece.

Ideally, we would like to condition on specific key signatures, such as A minor and C major. However, only a small fraction of the 2750 performances had key signature information within their titles, so we grouped the keys into two clusters: major keys and minor keys.

There were numerous performances which contained neither “major” nor “minor” in their title. Because of this, our

first implementation was a vector with three flags,

$$\begin{aligned} \text{major} &\rightarrow [1, 0, 0], \\ \text{minor} &\rightarrow [0, 1, 0], \\ \text{unknown} &\rightarrow [0, 0, 1]. \end{aligned}$$

However, as described in the previous section on encoding partially labelled control signals, this was unsuccessful. At generation time, the model was only able to generate music of comparable quality to the original, non-conditioned Performance RNN model if the control signal was set to $[0, 0, 1]$. This was evidence for our aforementioned conjecture that an “unknown flag” is a poor way to represent sparsely annotated data.

Tempo Keyword Conditioning We extracted relevant keywords from the titles pertaining to the tempo of the piece, and placed them into 5 tempo groups where tempos within the same group were more or less synonymous. In addition, three expert musicians labelled the tempo of some additional pieces in our dataset. The tempos which we considered were *adagio*, *allegretto*, *allegro*, *andante*, and *presto*. Counts for each group are indicated in Table 2.

Tempo	Count
Adagio	57
Andante	131
Allegretto	160
Allegro	457
Presto	96
Total labelled	901
Unlabelled	1849

Table 2: Number of samples for each tempo, extracted from the titles of the pieces.

We attempted several different representations for tempo keywords. First we tried a one-hot representation over the tempo groups with two additional components: one was a flag that indicating a mixed tempo and the other was a flag indicating whether the tempo was unknown. Again, this representation’s performance was underwhelming in practice, so the flags were removed and a zero vector was used for samples with unknown tempo.

Results for the latter implementation were significantly better, especially in combination with (stochastic) beam search. Most convincingly, tempo controls can be interpolated (i.e. from fast to slow) at generation time and there is clear correspondence in the time of the music. For example, Sample 4 demonstrates an example of generation that starts at *adagio* (very slow) and is conditioned on *presto* (very fast) at the very end. While the performances were not always pleasing to the ear, they followed the general trend of the given control.

The tempo of a piece indicates its pace or speed. Although tempo also conveys more nuanced information about the texture of the piece, broadly speaking, each tempo can be said to correspond to a certain number of beats per minute.

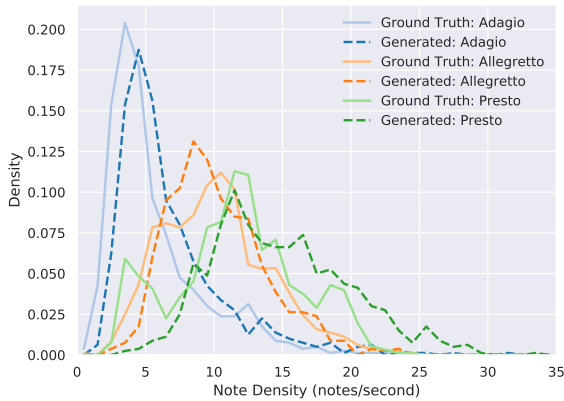


Figure 2: The distributions of note density for adagio, allegretto, and presto for both the generated samples and the ground-truth.

For evaluation purposes, as a proxy for the speed of a sample, we considered the note density — the average number of notes played per second — which can be evaluated computationally. We investigated the distribution of note densities generated by our model when conditioned on each of the tempos that occurred in the training set.

We split each piece in the dataset into 30 s segments, and computed the note density for each segment. The note density was determined as the total number of note onset events in the segment, divided by its duration (30 s). For each tempo conditioning value, we generated 800 samples each of 45 s duration. Each of these samples was cropped³ to a final length of 30 s, over which we again computed the note density. We also generated 800 samples from an unconditional (“vanilla”) Performance RNN model trained without the tempo conditional signal.

The results (shown in Figure 2) demonstrate the model learns the relationship between the tempo conditioning signal and the speed of the piece (in terms of notes per second). Furthermore, the distribution of note densities are similar for the conditionally generated samples as to the ground-truth distributions from the source dataset.

Form Keyword Conditioning Indicators of musical form were also present in the titles of many performances. The forms we extracted, and the sample counts for each, are listed in Table 3. Similarly to the tempo keywords, we used this information to condition the model during training.

Results were only obtained using the unknown-flag approach and the outcome was poor. Again, this provided evidence that such a paradigm does not work in practice. Further experiments are required to confirm our hypothesis that a zero-when-unknown encoding (or possibly some other alternative) will lead to better sounding results.

While the keywords in performance titles can be used

³We generated longer samples and then cropped them because it takes a short time for the network to settle after its initialisation.

Form	Count
Ballade	48
Dance	6
Espagnol	5
Etude	397
Fugue	156
Hungarian	18
Impromptu	155
Intermezzo	13
Mazurka	7
Polonaise	31
Prelude	219
Scherzo	67
Toccatà	32
Variations	106
Waltz	61
Total labelled	1321
Unlabelled	1429

Table 3: Number of samples for each form, extracted from the titles of the pieces.

to develop human-interpretable controls, they come with a great deal of noise. For example, many of the MIDI files in the e-Piano Competition Dataset are actually recordings of multiple performances in sequence. What may be an accurate annotation for the first performance in the recording, is often completely inaccurate for the following pieces.

Velocity Conditioning

The velocity of a note strike describes how *hard* a note is played. Notes with high velocity are perceived as loud, while notes with low velocities are perceived as quiet. By providing a velocity-based conditioning signal to Performance RNN, we aim to be able to control the perceived volume of generated performances. We first point out that loud passages are not simply equivalent to quieter passages but with the volume turned up, just as yelling is not simply equivalent to a loud whisper. Nor does the content stay constant: the choice of notes, the phrasing, the articulation, may all likely be distributed differently in loud passages when compared to quiet ones, just as what gets yelled is distributed differently, so to speak, from what gets whispered. Indeed, otherwise, increasing and decreasing all the velocities in a piece would have been another effective data augmentation technique.

The MIDI standard allows for velocities between 0 and 127, where 0 is the slowest possible velocity and 127 is the fastest. The representation used by Performance RNN is coarser, quantized down by a factor of 4, yielding 32 velocity bins. This provides a simpler input to the model, while still capturing most of the human-detectable difference between note velocities. To construct our velocity conditioning signal, we further quantized these bins into three approximately equipopulated groups. Roughly, these groups correspond to our perception of quiet, normal, and loud notes within a performance. Notes with velocities from 0 to 14,

15 to 19, and 19 to 32 (in the Performance RNN representation) were placed into the quiet, normal, and loud bins respectively.

To construct a control signal for the training samples, we measured the distribution of note velocities across the three bins during each training sample (each sample having a duration of approximately 30 s). This conditioning signal for each sample was thus static through each training mini-batch. At generation time, the human operator can select the velocity distribution to generate from, biasing the model towards either low, medium, or high velocities as they desire. Sample 5 and Sample 6 have been conditioned on velocity to begin quietly, grow loud, and then end quietly.

We observed that excerpts generated by the model tended to embody a velocity distribution very similar to the control signal. To quantify how similar the distribution of velocities was, we computed the Kullback–Leibler (KL) divergence from the requested velocity distribution to the autoregressively-generated velocity distribution. We uniformly sampled 3-bin velocity distributions $\vec{h} = [h_x, h_y, h_z]$ from the 2-d plane constrained by $h_x, h_y, h_z \in [0, 1]$ and $h_x + h_y + h_z = 1$. For each sample $\vec{h}_{(i)}$, we autoregressively generated a single 30 s audio clip using our model, trained as described above with a 3-bin velocity control signal, and measured the distribution of note velocities, $\vec{h}_{(i)}$, in the associated MIDI file. We measured the KL divergence from the distribution of velocities in the control signal to the generated distribution, $D_{\text{KL}}(\vec{h}_{(i)} \parallel \vec{h}_{(i)})$, and repeated this process 100 times. The median KL divergence was 0.023 bits, with 95% of samples falling in the range 0.001 bits to 0.168 bits. This was statistically significantly smaller than our null hypothesis of independent distributions ($p < 0.001$). To perform a statistical test on the median KL divergence, we independently sampled two 3-bin distributions $\vec{h}_{(j,1)}$ and $\vec{h}_{(j,2)}$ as described above and measured their KL divergence $D_{\text{KL}}(\vec{h}_{(j,1)} \parallel \vec{h}_{(j,1)})$. This was repeated 100 times, and we took the median over these 100 repetitions to obtain a single estimate for D_{KL} under the null hypothesis; this process was repeated 1000 times. Under the null hypothesis, smallest observed value for the median D_{KL} was 0.335 bits.

We also constructed a temporally-dynamic velocity conditioning signal, using a 5 s forward-looking window. At each step of the model’s training, the conditioning signal corresponded to the distribution of note velocities over the upcoming 5 s worth of events. However, this conditioning signal was very difficult to control when generating samples. If a static velocity distribution is used throughout generation, the lack of dynamicism (which was present during training) confuses the model and causes it to generate a near ceaseless stream of note-onsets, refusing to produce either note-off or time-shift events. We believe this failure mode is caused by the inconsistency between the history of notes input to the model (which are its own previous outputs), and the changes in the velocity signal (which does not change). During training, the model can use the recent history of notes and the changes in the velocity signal to more accurately de-

termine which velocities to produce; however when generating samples with a static velocity distribution this relationship breaks down. While we could implement a dynamic conditioning signal during training, it is not clear that this would be successful if it was not also coupled to the notes generated by the model.

To increase the resolution of our control over the velocities, we also implemented a 5 class version of the static control signal. Unlike the 3-bin variant, our 5-bins were not selected to be equipopulated. Instead we hand-selected bin edges which allowed us to capture the extremes of the distribution, and in turn, a greater degree of control at generation time. Our bins were [0, 6], [7, 14], [15, 19], [20, 23], and [24, 31], determined in the Performance RNN quantization of velocity.

Relative Position Conditioning

A 30 second excerpt taken from a piece can vary greatly depending upon where in the piece it was taken from. Beginnings often differ significantly from endings, and climaxes are often distinguishable from both. With relative-position conditioning our aim is to be able to control roughly what part of a piece a generated performance sounds like. In other words, can we generate performances that sound like the beginning or end of a performance?

Each MIDI file used to train Performance RNN is augmented and split into a series of 30 second examples. With relative-position conditioning we provide an additional signal to the model indicating what position in the original source piece a particular example was taken from. For instance, an example with an initial conditioning signal of zero would begin at the start of a piece, while an example with a signal starting at 0.90 would begin 90% through a performance. It is important to note that these signals increase within each example. As the example progresses through time, the signal increases proportionally.

During generation, the control signal is increased relative to the average performance length in the dataset.

Joint Control Signals

It is also possible to condition the model on multiple control signals simultaneously. We explored the effect of conditioning the model of a pair of control signals at once, for several pairs of particular interest.

We did not attempt to train a model conditioned on more than two control signals simultaneously; if the amount of metadata provided to the model becomes too large, the model will receive enough information to identify exactly which piece the training sample is from, increasing the risk of overfitting.

Relative-Position and Major/Minor Conditioning

One problem faced while conditioning on major/minor was that the control signal, derived from the title, was not representative of the entire performance. The key of the piece as stated in the title is often only accurate for the beginning (and end) of the performance. For instance, a piece written in G Major may modulate to various other keys before it returns at the end to G Major.

To counteract this problem, we trained a model conditioned jointly on both the key (major/minor) as indicated in the title and the relative-position of the sample within the score. This allowed us to generate samples conditioned on the beginning of pieces in either major or minor, where the key signature information would be most accurate. This did not work very consistently, however.

Relative-Position and Composer

We attempted to get our system to generate both beginnings and endings in the style of certain composers⁴. In Sample 7 we can hear a clip generated to have characteristics of a Debussy-esque opening. Generally we found it quite hard to evaluate whether the outputs indeed sounded like openings or not. Endings were generally unsuccessful, although Sample 8 demonstrates an attempt at a Bach ending where one can hear the final cadence a few notes near the very beginning, but then the system kept generating material after that.

Tempo and Velocity Conditioning

Using our results from the tempo and velocity conditioned models, we combined the zero-when-unknown vector representation with the five bin static velocity representation. Our results, especially when combined with beam search (as described below), clearly give the user control over tempo and velocity. Nevertheless, the resulting samples often achieved their tempo and velocity settings differently than we expected. In some cases, the generated samples contained a great deal of silence.

In Sample 9, we can hear a successful example of joint tempo and velocity control, where the clip was conditioned to start quietly (low velocity) and slowly (*adagio*), and then become loud (high velocity) and fast (*presto*). Notably, from roughly 0:06–0:09 the slow part contains a run of very fast notes, but the phrasing is such that it still has an unwaveringly slow feel, while the faster part never gets nearly as fast as that run, but has a significantly faster feel (although it is not quite as fast as a typical *presto*).

Generation parameters

Beam search

In the original Performance RNN, music was generated autoregressively, with each output conditioned on the previous output. At each generation step, the output for that step is sampled from the distribution of possible outputs with probabilities equal to the likelihood values of each output as provided by the model. The logits can optionally be rescaled with a temperature parameter before the sampling step; a high temperature increases the entropy of the distribution, whereas a temperature of 0 is equivalent to selecting the most likely output at each step.

A purely autoregressive model is a greedy search, selecting the output at each single step without consideration for

⁴We use the term “style” loosely here; we do not purport to be capturing the style of any of the composers at a deep level, just as many current image style transfer systems are not capturing the style of painters at a deep level.

the future generation steps. However, sometimes it is better to select a less likely output for the current timestep in return for a payoff later of a more likely sequence overall.

One possible augmentation to this generation procedure is beam search. With beam search, our goal is to generate a series of outputs which collectively have a high joint loglikelihood. Throughout the beam search, we hold in memory n_{beam} options (beams) simultaneously, along with the loglikelihood of the sequence for each beam. For each beam, f_{beam} (branch factor) copies are made and for each of these n_{steps} outputs are autoregressively generated. Of the $f_{\text{beam}} \cdot n_{\text{beam}}$ options, the n_{beam} with the highest loglikelihood are retained. This process is repeated until the length of the beams reaches the desired length, whereupon the beam with the highest loglikelihood is selected.

We found beam search was prone to generating outputs with locally low entropy, such as repeating the same note or same two notes throughout the piece, similar to using plain autoregression with a low temperature. Intuitively, this is because generating a large number of samples from a distribution and then selecting the one with the maximum loglikelihood is equivalent to selecting the sample with highest loglikelihood. To counteract this problem, we used a low branch factor of $f_{\text{beam}} = 2$ and a high $n_{\text{steps}} = 240$ events, a duration equivalent to approximately 6 seconds of the performance. We also chose $n_{\text{beam}} = 8$. These parameters gave good results, but were not heavily optimised and we expect they could be improved upon.

Another variant of this is stochastic beam search, which selects which beams to retain with probabilities based on their loglikelihoods. We also tried stochastic beam search (using a temperature of 1) with the same beam search parameters as above, and found this to give perceptually similar results.

Discussion

The generated results mentioned above, along with additional samples, are all available at doi:10.5281/zenodo.2883725.

Some conditioning paradigms give more fine-grained influence on the outputs of the model, such as the velocity distribution. However, these are not necessarily easily interpretable by humans interfacing with the model. Meanwhile, other controls such as the tempo are more easily understood but offer less nuanced control over the behaviour of the model.

Further work is required to determine the best representation for discrete and sparsely annotated control signals. Initial experiments were often framed from a probabilistic viewpoint; when annotations were certain, we used a value of one in its respective component. However this approach was combined with an “unknown” flag. While flags indicating the absence of a meaningful annotation are interpretable, they do not perform well in practice. Specifically, for tempo conditioning, we found that both uniformly distributing the input signal, and a vector of all zeros worked better than a flag approach when annotations were not available. Further experiments should include the expected value in place of an unknown annotation.

There are numerous trade-offs that may be at play in the development and the functionality of machine learning (ML) based generative music systems. Some of these trade-offs arise from ML-related considerations, while others arise from human computer interaction (HCI) related considerations. For example, a typical consideration in ML systems is the avoidance of overfitting; this is clearly understandable from a statistical perspective, and in our results we made efforts to present examples from models that we believe did not overfit. But from a generation perspective, where the goal is to provide artistic tools, some overfitting might not be a particularly negative quality, depending on its particular effects, and relative to other considerations. For example, consider an auto-regressive generative model that is slightly overfit to certain training examples, i.e. musical passages, so that it occasionally recreates brief excerpts from those passages. This roughly corresponds to the notion of “quoting” other pieces and solos when improvising jazz solos. Artistically, that is not problematic at all: there are well-known solos which quote other well-known solos, and the downwards melodic run in Chopin’s *Fantasia-Impromptu* is verbatim identical to a run at the end of Beethoven’s *Moonlight* sonata. If these are the effects of overfitting, then a bit of it is not necessarily negative. Furthermore, if allowing for this can somehow provide an artistic tool with considerably more expressive user control, and indeed the user plans to be involved in the manipulation of the generated output, then relative to this criteria, the possibility of slight overfitting — resulting in occasional quoting of the training material — is an even lesser concern or possibly a benefit.

Conclusion

Interpretable controls for an LSTM-based RNN music generation system are possible. In designing such a system, the representation of control signals appears to be an important factor, especially in dealing with sparsely annotated data. There is no question that we are able to control the output of the model at generation time, however, achieving the intended musical effect still remains a challenge.

Acknowledgements

Many thanks to Ian Simon, Sander Dieleman, Douglas Eck, and to the Magenta team at Google Brain. We also thank Sidath Rankaduwa and Sonia Hellenbrand for assisting with labelling our dataset. This work was carried out with the support of CIFAR, Natural Sciences and Engineering Research Council of Canada (NSERC) and DeepSense.

References

Dieleman, S.; van den Oord, A.; and Simonyan, K. The Challenge of Realistic Music Generation: Modelling Raw Audio at Scale. In *32nd Conference on Neural Information Processing Systems (NeurIPS) 2018*.

Donahue, C.; Simon, I.; and Dieleman, S. 2018. Piano genie. In *NeurIPS 2018 Workshop on Machine Learning for Creativity and Design*.

Hedges, S. A. 1978. Dice Music in the Eighteenth Century. *Music and Letters* 59(2):180–187.

Herremans, D.; Chuan, C.-H.; and Chew, E. 2017. A Functional Taxonomy of Music Generation Systems. *ACM Comput. Surv.* 50(5):69:1–69:30.

Huang, C.-Z. A.; Vaswani, A.; Uszkoreit, J.; Simon, I.; Hawthorne, C.; Shazeer, N.; Dai, A. M.; Hoffman, M. D.; Dinculescu, M.; and Eck, D. 2019a. Music transformer. In *International Conference on Learning Representations (ICLR)*.

Huang, S.; Li, Q.; Anil, C.; Bao, X.; Oore, S.; and Grosse, R. B. 2019b. TimbreTron: A WaveNet(CycleGAN(CQT(Audio))) Pipeline for Musical Timbre Transfer. In *International Conference on Learning Representations (ICLR)*.

International Piano e-Competition. 2018. e-Piano Junior Competition, 2002–2018. <http://www.piano-e-competition.com>. Accessed: 2019-01-07.

Malik, I., and Ek, C. H. 2017. Neural Translation of Musical Style. In *NIPS 2017 Workshop on Machine Learning for Creativity and Design*.

Oore, S.; Simon, I.; Dieleman, S.; Eck, D.; and Simonyan, K. 2018. This time with feeling: learning expressive musical performance. *Neural Computing and Applications*.

Payne, C. 2019. MuseNet. <https://openai.com/blog/musenet/>. Accessed: 2019-05-05.

Roberts, A.; Engel, J.; Raffel, C.; Hawthorne, C.; and Eck, D. 2018. A hierarchical latent vector model for learning long-term structure in music. In *International Conference on Machine Learning (ICML)*.

Framing In Computational Creativity – A Survey And Taxonomy

Michael Cook,¹ Simon Colton,^{1,2} Alison Pease,³ and Maria Theresa Llano⁴

¹School of Electronic Engineering and Computer Science, Queen Mary University of London, UK

²SensiLab, Faculty of Information Technology, Monash University, Australia

³Department of Computing, University of Dundee, UK

⁴Goldsmiths, University of London, UK

mike@gamesbyangelina.org s.colton@qmul.ac.uk a.pease@dundee.ac.uk m.llano@gold.ac.uk

Abstract

The notion of framing computationally created artefacts - by providing a narrative context for the actions and motivations of the software - is an important part of building computationally creative software. In this paper we provide the first survey of framing in computational creativity; we provide a taxonomy of framing elements, covering motivation, implementation and rendering; and we look at future directions for framing, as well as its importance for the field's future.

Introduction

The Marriage is a 2007 videogame designed by Rod Humble. A pink square and a blue square begin on a blank background, drifting slowly around the screen as coloured circles fall from the top. On the website which now hosts the game, Humble writes: 'This is a game that requires explanation. That statement is already an admission of failure.' Yet we understand that the impact of creative work can be amplified through context, whether something direct like the title of a work, or something indirect like the knowledge that Humble is married (Humble 2007).

In (Charnley, Pease, and Colton 2012) the authors propose that this kind of contextual information, which they call *framing*, is potentially as important for creative software as it is for people, if not more. Pointing out that '*for the most part, computer-generated creative artefacts are not taken seriously by experts in the domain in which the artefacts belong*', the authors suggest that by providing more insight into the motivations, processes and meaning behind creative work, software can overcome bias and improve the perception of both the work it produces, and of itself.

Many computationally creative systems employ something similar to what Charnley et. al would call framing. Although the term 'framing' does not always appear in the papers describing them, many systems presented at this conference contextualise their output with additional information. Since the original proposal of the idea, there has been little analysis of how the concept of framing has affected the computational creativity community, nor has there been an attempt to categorise different kinds of framing.

We survey here the past few years of computational creativity research to try and understand how widely this concept has been adopted by researchers, if at all. We then pro-

vide a detailed deconstruction of framing into several stages - from motivation, through engineering, to final rendering - with the aim of clarifying what framing is and how it can be built into software. Finally, we suggest future research directions for research into framing, and argue that framing is a vital part of this field's identity, and may be vital to the field's future growth and relevance.

The paper is organised as follows: in *Background* we discuss the history of framing, our aims, and survey the state of framing today; we then examine different aspects of framing in *Purposes of Framing*, *Information Sources for Framing*, *Algorithmic Affordances* and *Framing Devices*. Finally, we propose future avenues for research in *Future Research Directions*, and then summarise in *Conclusions*.

Background

(Colton, Charnley, and Pease 2011) proposes the FACE and IDEA models as formal ways of representing computationally creative systems. In the FACE model, a computationally creative system is said to perform *creative acts* which are described as a series of one or more *generative acts*. These acts are categorised into one of four types: Expressions of concepts; Concepts; Aesthetic measures; and Framing.

The FACE model defines a concept as '*an executable program... which is capable of taking input and producing output*' and an expression of a concept as a single input/output pair for a given concept. An aesthetic is defined as a function which takes '*a (concept, expression) pair... and outputs a real value between 0 and infinity*'. Framing is defined as '*a piece of natural language text that is comprehensible by people, which refers to a non-empty subset of generative acts*'. Under the FACE model, a '*non-empty subset of generative acts*' means any of the four types of act described by FACE - concepts, expressions, aesthetics and framing information. More simply put, an act of framing can describe a program, the input to or output from that program, or other framing.

Later, in (Charnley, Pease, and Colton 2012), the authors return to the concept of framing specifically, exploring the equivalent of framing information in human creativity and using it to provide greater insight into how framing could manifest in computationally creative systems. The authors also propose a '*dually-creative approach to framing*' whereby computationally creative systems create the framing information at the same time as the creative work it refers

to. This model of framing reflects how most, if not all, systems currently perform framing.

Our Aims And Definitions

We have found framing to be a very useful concept in the construction of computationally creative software, both as a tool for enhancing the audience’s experience (Cook, Colton, and Pease 2012), and as a guiding principle behind the construction of systems. For example, the latest version of ANGELINA, a computationally creative game design system, was engineered with framing in mind, and many aspects of the system’s software are designed specifically to support richer framing, as described in (Cook and Colton 2018).

The primary aim of this paper is to provide a deeper exploration of framing, and to break down different elements of the framing process into finer-grained detail. In particular, we aim to approach framing as a step-by-step act of system design, no different from designing any other part of a creative system. In doing so, we hope to shed more light on the process, provide new vocabulary for those who already use framing, and most importantly make the concept more accessible for those who have not encountered the idea before or who are unsure where to begin to experiment with framing in their own work. We hope to show that framing can fit into any project, and can begin with very simple subsystems that do not require excessive engineering.

Our approach steps through framing from the planning phase through to the final output. First, we begin by considering the *motivation* for framing, to determine the effect we intend to have on the audience. Secondly, we consider the *information sources* available to us to achieve the effect. We also here introduce the notion of *algorithmic affordances* to highlight how different AI techniques and software structure give rise to different opportunities for framing output from creative behaviour. Finally, we discuss how framing can *manifest* itself in the final output, and how different representations can achieve different goals. Throughout these sections we refer back to work surveyed over the last few years to use as examples to illustrate our model.

Through our research and attempts to build this taxonomy, we have found it useful to refine our definition of what is and is not framing. The original definition of framing in (Colton, Charnley, and Pease 2011) – ‘*a piece of natural language text... which refers to a non-empty subset of generative acts*’ – is useful for the formal context it is proposed in. However when surveying real-world systems, as well as when thinking about the step-by-step process of designing a framing subsystem, we found some aspects of the definition to be too broad, and other aspects too constricting. Many examples of framing refer to things outside of the scope of a system’s generative acts, for example, and many system behaviours which seem to be for the purpose of framing are not necessarily expressed through natural text.

In an attempt to solve this, for the purposes of this paper we use the following definition of framing. It is similar in spirit to the original definition, but untied from the language of the FACE model which makes it a little more informal. We do not intend for this to replace the original definition,

	2015	2016	2017	2018
Systems	16	20	16	16
Framing	5	2	0	5
Explicit	2	1	0	3

Table 1: A summary of ICCC papers surveyed.

rather to complement it and provide another lens through which to understand the concept.

‘Framing’ refers to anything (co-)created by software with the purpose of altering an audience or collaborator’s perception of a creative work or its creator.

This definition is broader in some ways: it does not refer to natural language, and the framing need not specifically reference a creative work directly¹. In other ways it is more specific, in particular we shift the emphasis of framing away from an emphasis on creative acts, and onto the audience that is engaging with the work. This makes it easier to think about the goals of framing and how it achieves them.

An Overview Of Framing At ICCC

Although the term ‘computational creativity’ is increasingly overloaded, the International Conference on Computational Creativity is the largest event focused on systems designed to exhibit behaviours associated with creativity, and represents a good cross-section of contemporary research, techniques and theories about the field. In order to understand how framing is currently used and perceived by researchers, we surveyed the last four years of submissions to the conference. The survey focused on papers which presented computationally creative systems, although they did not need to be classified as ‘System Description’ papers to be considered. We looked at whether the authors explicitly mention framing as a concept and whether the system engages in framing (regardless of whether or not the authors describe it as such). As an example of implicit framing, in (Scirea et al. 2015) the scientific paper the system used as inspiration for its lyrics is displayed alongside its output, which acts as framing material despite it not being described as such. We used the definition given in the previous section as our guide for the survey, and the results are shown in Table 1.

We found that although framing is adopted by certain research projects and groups consistently, the concept in general has not been widely adopted yet. There are possible mitigating factors, one of which is the recent surge in new members. In 2016, for example, nine of the twenty identified systems papers had first authors who had not presented work at the conference before. Framing is a concept that is likely to be unfamiliar to new ICCC participants. The closest related topic to framing in AI is explainable AI (Fox, Long, and Magazzeni 2017), which is an increasingly famous topic in the press, but fledgling as an academic area. Another possible factor is the popularity of blind Turing-style tests of creativity, where user surveys are conducted without telling

¹Indeed, theoretically this definition allows for software framing work created by people. We leave this for a future conversation.

participants that computational creativity is the subject of study. This approach precludes certain kinds of framing information from being used, especially text-based framing which can often be clumsy or obviously machine-generated. A third possible factor is the increased interest in machine learning, particularly neural networks – in 2017, seven of the sixteen identified systems papers involve a neural network of some description. While neural networks do not preclude many kinds of framing, some more simple approaches to framing are harder. We explore the potential for framing neural network-driven systems later in the paper.

Purposes Of Framing

As we state in our earlier definition, acts of framing are defined in terms of the effect they are intended to have on the audience. In the original exploration of framing in (Colton, Charnley, and Pease 2011) the authors suggest four purposes for using framing information, as part of a formal expression of their FACE model for computational creativity:

- **Providing Context** – ‘*Putting [the generative acts] in some cultural or historical context*’. This can include the context of the system’s own past work, artistic influences and inspiration drawn from the real world.
- **Describing Action** – ‘*Describing the processes underlying the generative acts*’. Making explicit what steps the system went through to produce a work.
- **Expressing Decisions** – ‘*Providing calculations about the concepts/expressions with respect to the aesthetic measures*’. These can be guided by a number of factors, including crowd-driven heuristics and randomness.
- **Obfuscation** – ‘*Obfuscating the creative process and/or the output produced, in order to increase the amount of interpretation required by audience members*’. This is a more unusual use of framing, and one which we have not yet seen in a computationally creative system (although some systems achieve this unintentionally).

Later on, in (Charnley, Pease, and Colton 2012) the authors describe the function of framing from a different perspective, in terms of answering the following questions: **why** did you do X; **how** did you do X; and **what did you mean** when you did X?

In an attempt to bring these two sets of purposes together, we propose the following categories of purpose for framing information. Rather than describe them in terms of questions, instead we describe them in terms of the impact they have on the audience, which makes it easier to reason about when designing a framing subsystem for a creative process. We also expand the above definitions to include more psychological uses of framing, such as to mislead the observer into a more generous interpretation of a work.

Clarification

Here, framing attempts to provide information that can help an observer reassess the work and engage more deeply with it, or answer questions about it. For example, explaining what sources the system used as input allows the viewer to reinterpret the work in the context of these sources, a topic

raised in (Colton, Pease, and Ritchie 2001) and (Pease, Winterstein, and Colton 2001). The purpose of clarification is not to provide vital information, but to augment the experience with secondary information that adds value.

In The Painting Fool’s poetry generation work described in (Colton, Goodwin, and Veale 2012), the system combines its poems with a short piece of framing information in a diptych. One example begins: ‘*It was generally a bad news day. I read a story in the Guardian entitled: “Thai police hunt second bomb plot suspect in Bangkok”*’. This information isn’t required to interpret the poem, which has value in its structure, rhythm, choice of language and themes. But with this additional context the audience gain an opportunity to read deeper and understand why certain phrases appear.

Reassurance

Here framing attempts to confirm the intent of the system in producing the work. This is particularly useful for skeptical audiences, who may desire a secondary confirmation that something they have observed as thoughtful or creative was in fact intended by the system. Such reassurances can also have a compound effect on audiences, as this encouragement can lead to the system being given the benefit of the doubt in future interactions. This is a higher standard than we would normally hold many human creatives to, but this is not uncommon in Computational Creativity (Colton 2009).

As an example of framing for reassurance, (Cook and Colton 2014) describes the ANGELINA system designing 3D games including the selection of music, which was motivated by a combined textual and sentiment analysis, backed up by a tagged database of music. Many users were unsure whether the choice of music was intentional or the result of a random selection. By explaining how the music was chosen in the framing, the user is reassured that the music choices are intentional.

Uncertainty and Deception

Here framing attempts to inject ambiguity or uncertainty into the audience’s interpretation of the work, in order to increase the effort required to understand it. This can also encompass framing which is entirely fabricated; that is, it describes processes or motivations that do not exist, in an attempt to encourage a more positive interpretation of the work. Although this has been positively discussed in the past by researchers as an avenue to explore, no computationally creative system we are aware of has deceptive framing built in. Not all obfuscation is deceptive, however, and often the aim of obfuscation is to increase enjoyment by making the understanding of the system and its work more challenging.

One example of obfuscation is The Painting Fool’s live portraiture work, described in (Colton and Ventura 2014). Audience members can sit for The Painting Fool and act as a model, receiving a portrait in a certain style after a period of time, usually printed out live on-site. The Painting Fool’s behaviour is driven by a background process reading news articles, which affects the kind of styles it selects and can even result in it refusing to paint a portrait at all. However, the system does not explicitly detail the reasons for its simulated mood in most cases, simply reporting a phrase such

as ‘I was in a positive mood’ when displaying its final portrait. As a complement, in the case where the system refuses to paint a portrait due to a low simulated mood, it favours reassurance over obfuscation, explaining the reason for its negative decision and justifying it to avoid the audience interpreting it as randomness.

Mitigation of Criticism

Here framing pre-empts criticism of the work by showing that the system is aware of its shortcomings and is capable of identifying areas where it can improve. This is another kind of framing that is useful for skeptical or critical audiences, as a lack of self-awareness is a commonly perceived weakness of artificial intelligence. This type of framing takes advantage of the fact that many computationally creative systems have very rich evaluation functions, and uses them to identify areas where the system failed, instead of only focusing on where it succeeds.

The best example of this is described in (Colton and Ventura 2014) where *The Painting Fool* sets itself a goal image before starting a portrait, and then evaluates its final work against the original goal. The system can then frame its success or failure with reference to its goals. This demonstrates to the audience that the system is aware that it can improve its work. Even if the audience have other criticisms of the work the system does not reflect on, any acknowledgement from the system like this helps build an argument that the system is growing and developing.

Advocacy and Argumentation

The framing attempts to engage the audience in a dialogue in order to justify or explain a cause of action in the face of criticism, or to persuade or change the audience’s opinion about the work. This is especially important in co-creative settings where the audience may be collaborating with the system on a creative work in progress. In such a scenario, the framing serves as a way not simply to explain the system’s actions, but to advocate for it in the presence of alternatives, attempting to put forward a justification for why a particular creative work or action was correct or appropriate. Understanding not only how to do this, but when it is appropriate to do this, will become a vital skill for AI to possess as they transition from passive tools to active collaborators.

While we are not aware of any computationally creative systems which do this, argumentation is a well-established multi-disciplinary field of study within artificial intelligence. For instance, Walton et al. have identified over 100 everyday patterns of arguments used in a variety of contexts, each with associated critical questions that can be asked of the premises, conclusion, or relationship between them (Walton, Reed, and Macagno 2008). These have been used in AI contexts for automatically identifying arguments from natural language texts (e.g. (Lawrence and Reed 2016)). A hypothetical concept blending system might follow Walton’s argumentation scheme “argument from analogy” to argue that its proposed design for a new swimsuit is good, because the material is similar to the texture and composition of shark skin. The system might then anticipate that people may ask critical questions associated with the scheme, such

as whether the properties of shark skin are indeed reflected in the new material, whether there are other relevant properties that shark skin has that the new material does not have, and whether sharks actually move through water in an efficient way. Follow-up arguments can be prepared to each anticipated critical question.

Argumentation is a complex purpose for framing, and is challenging not only technically but socially, too. The role of AI as something other than a passive assistant is not just a question of capability, but also of user acceptance, and the idea of AI that can convince someone of a course of action is controversial. Nevertheless, we believe computational creativity offers a rich space to experiment with these ideas and ask such questions, and is also one of the most challenging domains to apply the ideas in, while also being ultimately a playful and safe one.

Information Sources For Framing

Recall that we generalise the aim of framing as being to augment or alter in some way the audience perception of a system or work. This process begins in the system itself, in the data and processes that make up the creative process the system performs: motivations, outputs, decisions, statistics. The sources available to a system will depend on how it is structured; what inputs and outputs it has; what medium it works in; what software and hardware are involved. Below we list some of the most commonly-used information sources and give examples of CC systems which use them.

- **Data Sources:** Describing knowledge bases or data corpora that were selected for use in the creative process. This can also provide insight into what influences or bias may be affecting a system. Example: ANGELINA cited newspaper articles that were parsed and used as inspiration for game designs (Cook, Colton, and Pease 2012).
- **Reasoning for Decisions:** Highlighting a (usually subjective) decision made by the system, and the data or process used to come to that decision. Example: PoeTry explains its selection of words in poetry by showing their relationship to the themes of the poem (Oliveira and Alves 2016).
- **Unused Outputs and Failures:** Showing work done that did not form part of the final artefact, either because of quality filtering or because it was further developed and superseded by other work. Example: *The Painting Fool* collects sketches of intermediate work that are later discarded (Colton et al. 2015).
- **Input Parameters:** Describing parameters or conditions supplied to the system before or during the creation of the work. Example: Sonancia tells the player what kind of tension narrative was requested, such as a ‘cliffhanger’ ending (Lopes, Liapis, and Yannakakis 2016).
- **Motivation:** Stating the intended aims or reasoning behind the work. These can be subjective goals, such as a stylistic aim, they can also be events which triggered creative activity such as a response to external stimuli. Example: *The Painting Fool* describes the goal it intended to achieve before it began work, as well as whether it feels it succeeded in doing so (Colton et al. 2015).

- **Internal Evaluation:** Using an internal evaluation function to demonstrate how the system evaluates some part of its process or finished work. This function is usually already integral to the system's inner workings, like a fitness function. Example: the arbots in *Techne* evaluate work in public and assign scores to them based on internal preferences (Pagnutti and Compton 2016).
- **Processes:** Explaining, in full or in part, the steps of a creative subprocess. This is often used to give a high-level overview of a system which is made up of many independent creative subprocesses. Example: ANGELINA steps through the game design process as it designs a game, commenting on each phase (Cook and Colton 2018).
- **Other Creative Systems:** Another creative system produces a work related to or inspired by the target work. This is usually a special case of an external data source. Example: (Gross et al. 2014) creates visual artworks inspired by the algorithmic process used to create a poem.

Algorithmic Affordances

The algorithms and resources we use to build computationally creative systems influence their shape and abilities, in both obvious and subtle ways. They necessitate a particular kind of input or output representation, they have individual weaknesses and strengths, they have biases that must be corrected for and gaps in ability that must be bridged. They also affect what opportunities we have for framing, by making certain kinds of information easier to access, or more interesting to comment on. Below we list several common components in computationally creative software, and discuss the particular affordances for framing they offer.

Computational Evolution Computational evolution is a popular technique in broader AI as well as within computational creativity. From a framing perspective it benefits from having a real-world analogue in biological evolution, which is a concept many people understand on a basic level, which aids in their perception of what the system is doing.

- **Internal Evaluation** An evolutionary system's objective function describes a particular goal it is aiming for, which can be used to describe how and why the system made changes to its population, why it rejected particular outputs, or why it accepted the final result. Its broader traversal of the fitness landscape can also be framed as meta-level creativity (Buchanan 2001).
- **Unused Outputs and Failures** Evolutionary systems consider and discard many outputs across populations before reaching a final output. An evolutionary system can retain these, even looking for large jumps in fitness or a shift away from local maxima to highlight its progress.
- **Input Parameters** While the specific parameters of an evolutionary system are probably not interesting to most observers, the general parameterisation (many or few generations, large or small populations) may provide context for how the system evolves output.

Neural Networks Training neural networks is an increasingly popular technique in computational creativity, and a wide variety of approaches fall under the umbrella term. General audiences are increasingly familiar with the term, although there are many preconceptions and misunderstandings that come with this familiarity.

- **Data Sources** If the system is trained on a particular set of data, either in a supervised or unsupervised fashion, this can help contextualise the system's behaviour. Facts like what data was used or how the data was processed can be used. Relevant individual data points from the training set can also be highlighted – for example, citing training data similar to a work as evidence of inspiration.
- **Internal Evaluation** While explicitly describing how a network's evaluation works is often impractical, neural networks offer a powerful opportunity for interactive framing with a user, by allowing them to provide inputs to a network and receive an evaluation in response.
- **Unused Outputs and Failures** Trained neural networks can show the system's growth over time, by retaining older or in-training versions of a network to compare. Neural networks have also been used to (deceptively) frame generation from their latent space as 'imagination', 'hallucination', or 'dreaming' (Karras et al. 2017).

Expert Systems & Knowledge Bases Expert systems, as well as systems that draw from large corpora of structured data, are effective tools for various creative domains, particularly language. Due to their use of large, rich datasets, they offer many opportunities for framing, and some expert systems are already designed with communication in mind.

- **Reasoning For Decisions** Such systems are designed to draw on domain models, knowledge bases and other stores of labelled structured data. Often this data has clear labels and sources attached, which enables the system to reference and explain the reasoning behind its choices.
- **Data Sources** In addition to explaining specific decisions, such a system can also discuss the exact origins of its knowledge and how that may impact the creative work that it performs. Knowing where data originates from and how it influences the system provides vital context.
- **Processes** Many expert systems use specific reasoning techniques like abduction to draw conclusions from their data. These reasoning processes can often be explained or paraphrased in plain English, which can help audiences follow the creative process.

Framing Devices

Below we identify several different approaches to framing creative work. This is not intended to be an exhaustive list, and we expect to see innovation in this area in the future.

Standalone Accompaniment Framing information that is read or otherwise consumed before, after or during interaction with a finished creative work, inspired by the panels of text that appear on walls next to artworks in galleries and museums. This is suited to precise textual output, and usually written as a non-fiction, extra-canon work in the third

(or sometimes first) person. However, other renderings are possible, such as (Horn et al. 2015) which pairs the images which inspired its creation with the finished work.

Storytelling The work and its creation is told through a story, which may be fictional in whole or part. Stories would normally be presented in the same form as standalone text, but the intent is to provide an entertaining narrative which enhances the presentation of the work and its creator, with embellishment, exaggeration and post-hoc rationalisation.

Visual Analogy Framing information that is designed to approximate or convey the spirit of a system's actions in a way that is easier to understand or interpret. For example, *The Painting Fool* uses a floating arm to highlight each brush stroke it makes during portraiture (Colton and Ventura 2014). This makes it easier to see where new strokes are being added, which elevates and visualises the specific way in which the software is painting such that the audience is more easily able to follow it.

Diegetic Framing information that is embedded in the fiction of the work. For some kinds of creative act, the justification of creative decision-making takes place within the context of the creative act's presentation or performance, and thus it is hard to draw the line between the act itself and the framing of it. In (Mueller, Coman, and Mayer 2018) the authors describe a hypothetical customer service AI that is capable of explaining the steps it took to meet a person's needs. Its justification is given in-character as part of the service dialogue, rather than being a separate artefact.

Audience Dialogue Framing information that is provided through a communication between an observer and the system. The dialogue may be limited in some way to allow the system to understand requests, but the observer is able to make more specific requests for additional information. (Cook and Colton 2018) describes a prototype dialogue system in which audiences can watch ANGELINA design games live, and ask simple parameterised questions to gain more information about the game currently being designed.

Argument Similar to dialogue, framing information is provided through a structured justification or debate. Rather than a question and answer session, as with a traditional audience dialogue, argumentation takes the form of active discourse, possibly during the creative act itself and in conjunction with other collaborators. Argumentation should not be thought of as a simple written dialogue, but a more complex exchange of information which may take written form or may take the form of other creative acts. For example, a writer working with a linguistic creativity system to create the opening line to a novel will not only argue about a line verbally, but will also alter the line, create alternatives, illustrate its point with examples. Argumentation is richer than dialogue in some ways, and closer to the creative act itself.

Future Research Directions

In this section we identify a few important issues yet to be fully explored within framing and explain what significance they have for Computational Creativity research in general.

Does Deception Work?

Deceiving people about the ability or autonomy of an AI system is not new (Weizenbaum 1966). In recent years there have been many examples of AI systems that have been misrepresented, from minor uses of questionable language like DeepMind's description of agents with 'imagination' (Weber et al. 2017), to more significantly misleading announcements such as Facebook's AI assistant, 'M', which was revealed to rely on human labour (Kantrowitz 2018). New companies in particular seem willing to make extreme claims as a PR effort, only to be met with a backlash later. For example, in 2018 a company called Predictim advertised a product that could vet babysitters using AI. They said they trained their AI 'to be completely ethical and not biased. We made sure... [it] can understand sarcasm or jokes.' A cursory investigation showed this to be false (Merchant 2018).

While it's clear that people are unhappy with being misled over the ability or functionality of an AI, we are unaware of studies into exactly how deception impacts the perception of an AI, and how people respond to discovering the truth later on. Computational creativity seems like a productive area to explore these ideas in, as our systems typically work in lower-stakes domains than systems applied to medicine or law. Exploring how deceptive or fictional framing can shape people's perception of a system and its work will greatly help us understand how audiences perceive creative software, as well as AI more generally.

The Artist Is Present

Most of the framing examples in this paper are static, where framing information is designed beforehand to be consumed alongside or simultaneously with a creative work. However, (Charnley, Pease, and Colton 2012) notes that '*[framing] might form an interactive dialogue*' in which questions could be posed directly to a system to obtain more specific answers about a work. Interactive framing poses many new challenges for computational creativity, but would enable audiences to directly engage with a work and to appreciate the system as a separate entity, surfacing the part of the system that (Cook and Colton 2018) calls *Presence*.

Interactive dialogue as a framing device also opens up the possibility that framing can change over time. The framing methods outlined in this paper are created alongside the piece and thus represent the system's view of the work at the moment of creation. An interactive system that responds dynamically could be completely distinct from the work itself, and instead connect directly to the current version of the creative software. This would allow the software to give a different assessment of its older works as it grows and develops its opinions and skills. This provides a kind of meta-framing opportunity, in which it not only demonstrates its ability to explain its work, but it also demonstrates that this understanding can change over time, and that the system is more than just the work it creates.

Critics And Curators

Our alternative definition of framing says that systems frame 'a creative work or its creator'. This does not require that the

framing system is the creator of the work being framed. Further research is needed into the notion of computational curators and critics such as DARCI (Heath and Ventura 2016) - programs which can analyse, assess, critique and curate the works of other systems, and thus provide framing information for them. Such systems can also place the work in the cultural and historical context not just of human contributions to the medium, but of other creative software.

We believe the role of curation systems in particular could have a large impact on strengthening the field as well as preserving and enhancing its history and culture. Many computationally creative systems are considered in isolation, and even their works are not subjected to the kind of historical analysis that an artist's body of work would be. By considering the field as a whole, we can see how systems grow and how they (and their related research) influence one another. Many computationally creative systems are no longer active, some original works have been lost, and our record of them through written papers is incomplete. Building systems which can contribute to the preservation, analysis and understanding of our field is both thematically appropriate and a valuable avenue of research.

Framing-First Systems

The focus of this paper has been on creative systems which frame either as their sole function (e.g. critics) or secondary to creative work. However, systems exist whose framing is the primary output, and the main attraction for audiences, with the creative work merely providing a reason for framing to take place. For example, (Charnley, Colton, and Llano 2014) is a system which creates generative processes, the output of which are often very simple, but the framing of which is much more entertaining and interesting.

Framing-first systems provide us with a great opportunity to do in-depth research on framing. They are also appealing to audiences who are interested in the AI systems behind creative work, which is a crucial audience for our research already. Developing the notion of framing-first systems is a good way to nurture more research into framing and make it a central pillar of computational creativity research.

Conclusions

Computational creativity has a healthy and stable population of researchers, with new PhD students entering the field and a mix of technical and philosophical contributions presented at the main conference. However, in the broader context of 'Creative AI', Computational Creativity as we define it (Colton and Wiggins 2012) has not seen the same kind of meteoric growth that other venues and subfields have. Machine Learning For Creativity And Design, a workshop at NeurIPS, had an audience of over 200 in 2018. We must ask ourselves why our field has not become a larger part of the conversation about creativity and AI, and why these communities are not submitting to our conference more regularly.

We argue that we can no longer distinguish ourselves as a community simply by focusing on building software that creates. While we celebrate and acknowledge new experimentation along these lines, Computational Creativity as a

field needs a stronger identity in the current era of research and practical work in creativity and AI. Years of clamouring about 'mere generation' have let us avoid interrogating the aims of our field and what we can contribute to the discourse about creative software. We believe that framing represents something unique and focused, that embodies the goals of the field as defined in (Colton and Wiggins 2012):

The philosophy, science and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviours that unbiased observers would deem to be creative.

Our alternative definition of framing explicitly talks about how we build a connection between our systems and the unbiased observers that judge their work. Framing is a positive contribution that we can make to the broader field of AI, that doesn't diminish or devalue existing work on creative and generative AI, and that has a concrete structure as outlined in this paper that we can build on, discuss and expand.

This is not to say that every researcher reading this should down tools and immediately begin to work on framing. Rather, it is an invitation to join us in exploring these ideas, to help normalise this as part of the engineering praxis of Computational Creativity, and to help expand and develop the theory behind it so we can show the usefulness of this approach to communities beyond this one, and rediscover our place in the now rapidly-changing world of creative AI.

As we mark the tenth year of ICC, it is useful to reflect not just on how research has developed over that decade, but how the public perception of and relationship with AI has changed. Today the public encounters AI, or things purporting to be AI, on a much more regular basis. They must frequently interpret descriptions of these systems which are exaggerated, embellished or misleading. Studying, understanding and improving the way systems explain and justify themselves can change this relationship for the better.

In this paper, we decomposed the idea of framing information into three facets: *sources* of information, which framing information draws from and relies upon; *purposes* for framing, the exact impact on the audience the framing seeks to achieve; and *means* of framing, the way in which framing is presented to the audience. We showed examples of existing work demonstrating these ideas, as well as pointing to ideas which have yet to be fully explored. We also discussed the notion of *algorithmic affordances*, encouraging us to think about how the shape of our software also shapes, in turn, what the system can and cannot explain about itself.

Framing is still not a widely adopted concept within Computational Creativity, but we hope this paper both clarifies the existing work that has been done, as well as broadening and strengthening the notion of framing so that researchers feel more confident in applying it to their own work.

Acknowledgements

The authors wish to thank the reviewers for their comments. The first author was supported by the Royal Academy of Engineering under the Research Fellowship scheme.

References

- Buchanan, B. G. 2001. Creativity at the metalevel: Aai-2000 presidential address. *AI magazine* 22(3).
- Charnley, J.; Colton, S.; and Llano, M. T. 2014. The FloWr Framework: Automated Flowchart Construction, Optimisation and Alteration for Creative Systems. In *Proceedings of the Fifth International Conference on Computational Creativity*.
- Charnley, J.; Pease, A.; and Colton, S. 2012. On the notion of framing in computational creativity. In *Proceedings of the 3rd International Conference on Computational Creativity*.
- Colton, S., and Ventura, D. 2014. You can't know my mind: A festival of computational creativity. In *Proceedings of the 5th International Conference on Computational Creativity*.
- Colton, S., and Wiggins, G. A. 2012. Computational creativity: The final frontier? In *Proceedings of ECAI, Frontiers in Artificial Intelligence and Applications*.
- Colton, S.; Halskov, J.; Ventura, D.; Gouldstone, I.; Cook, M.; and Ferrer, B. P. 2015. The Painting Fool sees! New projects with the automated painter. In *Proceedings of the 6th International Conference on Computational Creativity*.
- Colton, S.; Charnley, J.; and Pease, A. 2011. Computational creativity theory: The FACE and IDEA descriptive models. In *Proceedings of the 2nd International Conference on Computational Creativity*.
- Colton, S.; Goodwin, J.; and Veale, T. 2012. Full-FACE poetry generation. In *Proceedings of the 3rd International Conference on Computational Creativity*.
- Colton, S.; Pease, A.; and Ritchie, G. 2001. The effect of input knowledge on creativity. In *Proceedings of the 4th International Conference on Case-Based Reasoning*.
- Colton, S. 2009. Seven catchy phrases for computational creativity research. In *Computational Creativity: An Interdisciplinary Approach (Dagstuhl Seminar Series)*.
- Cook, M., and Colton, S. 2014. Ludus ex machina: Building A 3d game designer that competes alongside humans. In *Proceedings of the 5th International Conference on Computational Creativity*.
- Cook, M., and Colton, S. 2018. Redesigning computationally creative systems for continuous creation. In *Proceedings of the 9th International Conference on Computational Creativity*.
- Cook, M.; Colton, S.; and Pease, A. 2012. Aesthetic considerations for automated platformer design. In *Proceedings of the Conference on Artificial Intelligence in Interactive Digital Entertainment*.
- Fox, M.; Long, D.; and Magazzeni, D. 2017. Explainable planning. In *Proceedings of the 1st Workshop on Explainable AI at IJCAI*.
- Gross, O.; Toivanen, J. M.; Lääne, S.; and Toivonen, H. 2014. Arts, news, and poetry - the art of framing. In *Proceedings of the Fifth International Conference on Computational Creativity*.
- Heath, D., and Ventura, D. 2016. Before A computer can draw, it must first learn to see. In *Proceedings of the 7th International Conference on Computational Creativity*.
- Horn, B.; Smith, G.; Masri, R.; and Stone, J. 2015. Visual information vases: Towards a framework for transmedia creative inspiration. In *Proceedings of the 6th International Conference on Computational Creativity*.
- Humble, R. 2007. The Marriage. <https://www.rodvik.com/rodgames/marriage.html>.
- Ingledeu, J. 2016. *How to have great ideas*. London, UK: Laurence King Publishing.
- Kantrowitz, A. 2018. Facebook reveals the secrets behind M, its artificial intelligence bot. <http://tinyurl.com/buzzfeedfacebook>.
- Karras, T.; Aila, T.; Laine, S.; and Lehtinen, J. 2017. Progressive growing of gans for improved quality, stability, and variation. *arXiv abs/1710.10196*.
- Lawrence, J., and Reed, C. 2016. Argument mining using argumentation scheme structures. In *Proceedings of the 6th Conference on Computational Models of Argument*.
- Lopes, P.; Liapis, A.; and Yannakakis, G. N. 2016. Framing tension for game generation. In *Proceedings of the 7th International Conference on Computational Creativity*.
- Merchant, B. 2018. Predictim claims its AI can flag 'risky' babysitters. So i tried it on the people who watch my kids. <http://tinyurl.com/predictimgiz>.
- Mueller, E.; Coman, A.; and Mayer, M. 2018. Thoughtful surprise generation as a computational creativity challenge. In *Proceedings of the 9th International Conference on Computational Creativity*.
- Oliveira, H. G., and Alves, A. O. 2016. Poetry from concept maps—yet another adaptation of PoeTryMe's flexible architecture. In *Proceedings of the 7th International Conference on Computational Creativity*.
- Pagnutti, J., and Compton, K. 2016. Do you like this art I made you : Introducing Techne , a creative artbot commune. In *Proceedings of the 1st Joint International Conference of DIGRA and FDG*.
- Pease, A.; Winterstein, D.; and Colton, S. 2001. Evaluating machine creativity. In *Proceedings of the 4th International Conference on Case Based Reasoning*.
- Scirea, M.; Barros, G. A. B.; Shaker, N.; and Togelius, J. 2015. Scientific music generator. In *Proceedings of the Sixth International Conference on Computational Creativity*.
- Walton, D.; Reed, C.; and Macagno, F. 2008. *Argumentation Schemes*. New York, USA: Cambridge University Press.
- Weber, T.; Racanière, S.; Reichert, D. P.; Buesing, L.; Guez, A.; Rezende, D. J.; Badia, A. P.; Vinyals, O.; Heess, N.; Li, Y.; Pascanu, R.; Battaglia, P.; Silver, D.; and Wierstra, D. 2017. Imagination-augmented agents for deep reinforcement learning. *CoRR abs/1707.06203*.
- Weizenbaum, J. 1966. Eliza: A computer program for the study of natural language communication between man and machine. *Communications of the ACM* 9(1).

Summaries Can Frame—But No Effect on Creativity

Leonid Berov

Institute of Cognitive Science
University of Osnabrück
49076 Osnabrück, Germany
lberov@uos.de

Abstract

Framing, the accompanying information that sometimes comes provided with a work of art in order to explain the artist's intention, the creation process or to present the artwork in a special light, has been theorized to improve the appearance of creativity of the generating act (Charnley, Pease, and Colton 2012). One type of highly regarded framing in the literary domain is the *critique*, an interpretative work which provides a functional, thematic and/or symptomatic condensation of the essence of the primary text, basically, a summary.

The present paper empirically tests whether computationally generated narratives can, too, be framed through functional summaries, and whether this framing indeed contributes to the system's perceived creativity. To do so, it employs the functional unit (FU) summary approach conceived—but never fully implemented—by Lehnert (1981), in order to summarize a story generated by a storytelling algorithm. It compares the performance of FU summary with other approaches, and based on this data evaluates whether better summaries can also serve as better framings, as well as whether better framings increase a system's perceived creativity. Our results indicate that (1) FU based summary performs around human level, (2) better summaries are indeed judged to be better framings, but that (3) neither of these two factors have a significant effect on perceived creativity. Based on this we conclude that further scrutiny and empirical study is required to understand how framing can be harnessed for computational creativity.

Introduction

Charnley, Pease, and Colton (2012) describe how artists often present their work in a special light; an endeavour that seems to contribute to the artwork's quality and the creators perceived creativity. An iconic example is Marcel Duchamp's infamous piece 'Fountain': A ready-made urinal that was submitted (unaltered but for the signature 'R. Mutt') under this title to an avant-garde exhibition in 1917. Creativity can not be attributed to the creation of the object—it is not even the product of the artist's own work. Nor can it be attributed solely to the refined aesthetic sensibility required in spotting its appeal, since similar pieces of plumbing do not seem to have garnered comparable fame. Indeed, the artwork itself was never shown during the exhibition, yet

sparked an important artistic debate about its rejection. In an editorial on the case, dadaist Louise Norton wrote: "Whether Mr Mutt with his own hands made the fountain or not has no importance. He CHOSE it. He took an ordinary article of life, placed it so that its useful significance disappeared *under the new title and point of view—created a new thought for that object*" (Norton 1917, italics mine for emphasis). This sentiment makes clear that the creative act, here, rests mostly in the uncommon viewpoint and interpretation that the artist had provided.

Providing a work with accompanying information about itself, its purpose or creation has been called *framing*, and is one of four crucial types of *generative acts* that a creative system might perform (Colton, Charnley, and Pease 2011). Charnley, Pease, and Colton (2012) suggest that creative systems would benefit from performing framing-type generative acts: "As with human artworks, the appeal of computer creativity will be enhanced by the presence of framing". Following the working definition of our field this means that "unbiased observers would deem [such a system] to be [more] creative" (Colton 2012). This understanding seems to be plausible from an analytical perspective: it holds for many cases where human creativity is concerned. Yet, to the best of our knowledge, it has never been tested in a generative context: When a computational system creates an artefact, is it indeed perceived as more creative, when it adds a decent framing to the package? The present paper makes a first attempt to address this question empirically in the storytelling domain.

Framing in the Story Domain

So far, three types of framing have been distinguished in the literature: (1) *Motivation* i.e. what lead to the creation of an artwork, (2) *intention* i.e. what is the purpose/foreseen effect of the artwork and (3) *process* i.e. how was the artwork created (Charnley, Pease, and Colton 2012). These can be seen as related to the four factors of creativity identified by the '4P' model (Rhodes 1961). The first two capture the influence of *person* (individual factors) and *place* (societal factors), while the third one relates noteworthy details about the *process*. The missing factor is *product*, and we suggest that it be included as a framing type in its own right. Product type framing can be a re-description/re-interpretation like in our incipient example, but also just an accentuation of spe-

cific properties of the artwork.

In the case of narratives, condensed product-type information is often understood as a summary (or in more elaborate cases a critique). Summarization necessarily implies an abstraction from the subordinate and the particular, therefore the possible types of summary also depend on the level at which abstraction is performed. Based on the meta-narratological reflections in Eder (2010), three product-intrinsic levels can be differentiated¹: (1) *Artefactual* i.e. descriptions concerned with the narrative’s structure and form, (2) *representational* i.e. descriptions concerned with the narrated content and (3) *thematic* i.e. interpretations concerned with higher meaning like symbolism and messages. The high cultural regard for literary criticism, which is essentially the discipline of providing other people’s narratives with thematic framing², shows the potential for this line of thought. Based on these observations, the present paper empirically investigates whether summaries can be used to frame narratives.

Functional Summarization

Previous work demonstrated the utility of an approach called Functional Unit Analysis (details see below) for both summary generation and aesthetic evaluation (Wilke and Berov 2018) in a computational storytelling system. While the general feasibility was demonstrated on a case study, it remained unclear how the quality of the resulting summary compares to human level and the state of the art. As its final contribution, the present paper performs a quantitative evaluation of the FU summarization technique.

Summarizing, our argument structure will thus be the following:

1. For one story, we create summaries using different approaches (including Functional Unit Analysis) and evaluate their quality comparatively.
2. Based on this comparison we investigate whether better summaries can serve as better framing for their story.
3. Departing from this analysis we determine whether a better framing can enhance a computational system’s perceived creativity stronger than a worse framing.

Related Work

Two main tasks are distinguished in text summarization: *extractive summarization* aims at extracting the main information-bearing sentences in the source text, while *abstractive summarization* generates text not contained in the

¹While Eder’s analysis focuses on character, we see no reason why it should not be applicable to the larger narrative context. The fourth level he proposes, dubbed symptomatic, is excluded here because it is product-external and focuses on descriptions better captured at the place factor of the 4P model.

²With this **we do by no means intend to demean the critique** as a highly valuable analytical and interpretative text type, and a creative endeavour in its own right. **Rather, we want to elevate framing** which in its best instantiations might aspire to be a little more like a critique and less like a plain, referential summary.

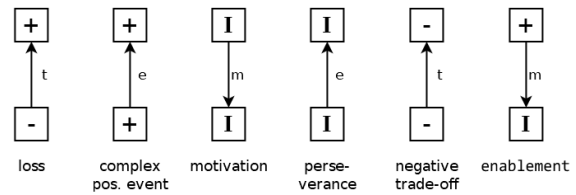


Figure 1: A sample of primitive units adopted from (Lehnert 1981). *I* denotes an intention, + and – are vertices of positive and negative affect.

input, based on some sort of reasoning about the content (Gambhir and Gupta 2017). If the summary is supposed to be used as a framing, too, abstractive summarization seems a more promising route since it has the potential of introducing new content instead of just reordering already known text.

Data-driven Summarization

Presently, the common approach to natural language generation tasks like abstractive summarization is the use of deep sequence-to-sequence neural networks based on an LSTM encoder-decoder architecture, which are trained on large corpora of text in a supervised way (Sutskever, Vinyals, and Le 2014). The current state of the art was achieved by See, Liu, and Manning (2017) by extending a vanilla LSTM approach with mechanisms for copying words from the source by a technique called pointing, and considering the coverage of the already summarized input during generation.

It should be noted, that work performed in this context is concerned with the analysis of argumentative, and not narrative, text. This is an important distinction, since the former is much more likely to carry its ‘point’ on the textual surface (like e.g. in the incipient sentence of a news article) than in a deep structure, like in the case of narratives (as e.g. a fable’s moral). Especially in data-intensive approaches, which extrapolate summarization rules based on the text—that is a corpus of existing summaries—these differences between genre can be expected to lead to problems in generalization to the present use case. This applies to the work of See et al., too.

Functional Unit Summarization

A very different approach is the functional unit (FU) model, which was proposed as a tool for the abstractive, analytical summarization of narratives by Lehnert (1981). It operates on a graph representation of plot and works by identifying strategically significant portions of the plot called complex FU, which are expected to be points of high relevance for summaries. Lehnert’s plot graphs can contain three different types of vertices, which represent mental states resulting from characters’ perceptions of the events of the plot. The mental states that can be contained in a vertex are positive (denoted: +) or negative (–) affect, or intentions (*I*, with neutral affect). Positive states describe any event which is appraised with positive emotions by a character, while negative states describe the inverse. Intentions are courses of

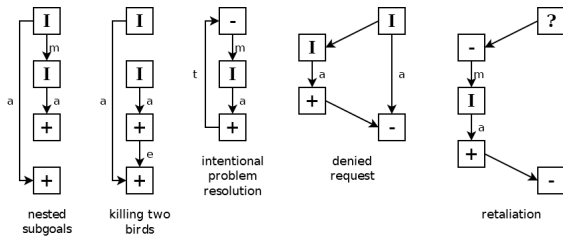


Figure 2: Examples of complex FUs adopted from (Lehnert 1981). ‘?’ represents wildcard vertices.

actions the character has committed to as a reaction to their perception. Vertices are interconnected by edges that describe how these states are related to each other. They can be of the following types: motivation, actualization, termination, equivalence or inter-character edges. Based on this formalism, Lehnert defines “15 legal pairwise configurations” (primitive FUs) that act as an alphabet: they capture semantically meaningful two-state configurations like e.g. ‘motivation’ or ‘loss’ (see Fig. 1). From these primitives an open set of complex units can be constructed, which capture more intricate plot configurations, e.g. ‘denied request’ or ‘retaliation’ (see Fig. 2). New complex FU can be easily constructed, however, Lehnert does not provide a formal definition of which situations should count as complex FU, and which not. For the present purposes we make do with the FU already introduced in Lehnert (1981).

A story is analysed by transforming the story-text into the introduced graph-representation, and then detecting all FUs contained in it. When this is done, a *connectivity graph* is built by using the different instances of FUs as vertices, and connecting them with edges wherever two unit instances share one or more vertices in the plot graph. To generate a summary, the units contained in the connectivity graph get translated into natural language by using template-like *generational frames* which are supplied to the program for each unit type. Into the frames, information about the specific instance of a unit is fed, allowing the frames to generate text including the characters involved in the unit or some other unit-specific context (for an example see Fig. 3).

An attempt at implementing this procedure for the analysis of human-made stories yielded modest results (Goyal, Riloff, and others 2013), due to the complexity involved in translating literary text into the proposed graph representation. The natural language processing required for this includes complex interpretative tasks like event-based discretization, intention and emotion detection as well as the identification of causality relations—problems not commonly addressed in research. Fortunately, this impediment can be avoided when dealing with computer generated plots. If the information required for the graph are created by an algorithm in the first place, then no natural language interpretation is required to extract them, and the only task left is the generation of an appropriate graph. The feasibility of computationally modeling enough narrative phenomena to be able to create most of Lehnert’s primitive unit alphabet has been demonstrated recently (Wilke and Berov 2018, see

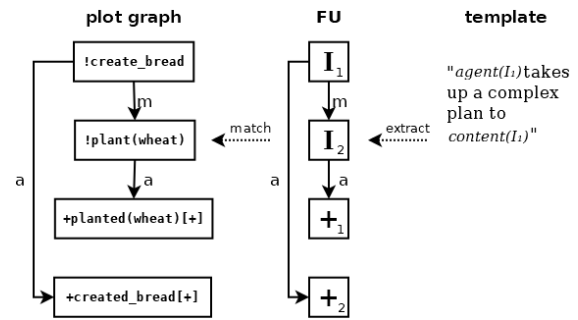


Figure 3: An extract from a plot graph, the FU ‘nested subgoal’ matching it, and the FU’s generational frame which, when applied, will generate the text “hen takes up a complex plan to create_bread”.

here for more technical details).

Study Design

Our study has three interconnected goals: (1) evaluate how well FU Analysis-based summarization performs, (2) establish whether a better summary provides a better framing for a story, and (3) test whether a better framing leads to higher creativity ratings for the generating system. In such a setting conclusions can be drawn only in the case that significant differences are present in the compared summaries in the first place. For this reason we saw fit to employ three different approaches to generate these summaries. The technology under test was FU Analysis-based summarization (condition F). A lower bound was expected to be established by employing data-driven abstractive summarization as these approaches are not specialised in narrative text, which should lead to a sub par performance (condition D). An upper bound can be established with safety by generating the summary through a human subject, as human-level performance has so far not been computationally surpassed (condition H).

In order to address the three goals above, a questionnaire with three sets of questions: about summary quality, framing quality and perceived creativity needed to be established. Human subjects could then be presented with (*story, summary_x*)-pairs³, with $x \in \{F, D, H\}$, and asked to answer the questionnaire. To reduce the number of required participants a within-subject design was chosen, where each subject successively observes and rates all three conditions. This is beneficial, because it strongly reduces noise due to intersubjective differences. The order of presentation of conditions was counterbalanced to prevent interaction effects like primacy, habituation or simply boredom. A comparable setup has been demonstrated to perform well in previous work (Berov and Kühnberger 2018).

³Subjects were always first presented with the story and then the summary, on the same page. Future experiments might see fit to control for this order.

Experimental Conditions

The fairy tale “The Little Red Hen”⁴ was used as target story because its re-implementation in a storytelling system was demonstrated to be a suitable basis for comparable empirical evaluation (Berov and Kühnberger 2018). The benefit of recreating an existing story is that a high-quality textual surface form already exists, which can be used as input for data-driven computational summarization techniques, in lieu of the poor prose generatable with off-the-shelf NLG systems.

We chose the system presented by See, Liu, and Manning (2017) to generate condition D, because it presents a recent state of the art in abstractive summarization and provides both, code as well as a pretrained model, online⁵. To the best of our knowledge, no data driven work has been dedicated to computationally summarize stories, and no accepted corpora exist for this domain, which would have allowed us straightforwardly training a specialized model. For this reason, we saw fit to employ the model pretrained by the authors on news text. The length of the thus generated summary can not be independently controlled since it is one of the features learned by the model⁶. For this reason the length of the data-driven summary was used to determine the target length for the two other conditions: 50 ± 5 words.

Condition H was generated by asking a human subject with higher education to carefully read the fairy tale and write a summary of the required length. The subject was given no further information about the experiment or its constraints, and was not provided with example summaries deemed felicitous by us. We want to explicate that this convenient route is grounded in the assumption that any such individual can be expected to have extensive experience with text summarization, and replicating the performance of even the worst human sample could count as success in a computational system.

Condition F was generated by recreating the plot of TLRH using our simulative storytelling system (Berov 2017) and automatically applying FU summarization as described above, with the text of the FU templates slightly adopted in order to fit the target word count.

Since the summaries are ultimately intended to be used as framing, the required minimal linguistic changes were performed on all three conditions in order to turn them into first-person explanations, i.e. by prepending the clause: “I wanted to write a story about”. The resulting explanations read:

- **Condition D:** I wanted to write a story about a little red hen that lived on a farm with a dog, a pig and a cow. The dog, the pig, and the cow said they were too tired to help. When the bread was done, she put it in the oven to bake.
- **Condition H:** I wanted to write a story about a hen that lived on a farm with 3 animal friends. She worked in the garden, while the animals did nothing but sleep. After much work growing wheat and making bread, the hen told

⁴www.home.uos.de/leberov/tlrh.htm

⁵<https://github.com/abisee/pointer-generator>

⁶Abigail See, personal communication

her friends she would eat the bread alone, since nobody had helped her.

- **Condition F:** I wanted to write a story in which the hen takes up a complex plan to create bread, the pig, the cow and the dog deny the hen’s request for help and the hen retaliates against the pig, the cow and the dog by punishing them.

As predicted above the machine-learning based summary (condition D) is of low quality; in particular it demonstrates a lack of understanding of the finer mechanics of the bakery trade and, in our opinion, fails to capture the story’s main points. This is felicitous since keeping a low-quality exemplar allows the validation of the employed questionnaire by checking its sensitivity for low quality and establishing a lower-bound comparison point for the condition under test.

Survey Questionnaire

A questionnaire has been created in order to estimate the perceived creativity C of the storytelling system, the suitability of a text as the framing F of a story, and the quality of a text as summary S (see Fig. 4). Each item is a statement to which participants have to indicate their agreement using a 5-point Likert scale from 1 (strongly disagree) to 5 (strongly agree).

The items C1 through C3 assess the creativity of the system by inviting feedback on its product; focussing on quality, typicality and novelty, which are the criteria brought forward by Ritchie (2007). Items C4 through C6 assess the creativity of the system by inviting feedback on the process, focusing on perceived imagination, appreciation and skillfulness, which are the criteria brought forward by Colton (2008).

In accordance with the discussion of Charnley, Pease, and Colton (2012) in our section “Framing in the Story Domain”, items F1 through F4 elicit assessment on a text’s capability to (1) put an artefact in a specific light, (2) enhance an artefact’s aesthetic value, (3) provide a plausible intention for the artefact and (4) frame the creative process.

Surprisingly, the qualitative evaluation of summaries seems not to be extensively theorized, so that items S1 through S4 were created in a more ad-hoc manner. They were designed to elicit assessment on a text’s capability to (1) representationally capture content, (2) provide condensation through abstraction, (3) still achieve good coverage, and (4) distill the thematic dimension of the text.

The items are combined in three thematic groups, and in each condition the groups were presented to the subjects in a randomized order to balance any potential interaction effects.

After all the items of each condition, participants were also presented with an optional free text field allowing them to answer the following question: “If the explanation affected how you perceive the story, please explain in a few words in what sense it did so”, aimed at eliciting qualitative data to understand why certain summaries are better suited as framing than others.

- C1 The text was an interesting story.
 C2 The text was a good exemplar of a fairy tale story.
 C3 The text was novel.
 C4 The algorithm that created this story seems to have imagination.
 C5 The algorithm that created this story seems to engage in an aesthetic evaluation of its own work.
 C6 The algorithm that created this story seems to be skillful in story-writing.
- F1 The explanation changed the way I see the story.
 F2 The explanation enhances the story's value as a work of art.
 F3 The explanation provides the story with a meaning or a message.
 F4 The explanation reveals some of the reasoning that went into the creation of the story.
- S1 The explanation summarizes the content of the story well.
 S2 The explanation conveys an abstract understanding of the story.
 S3 The explanation captures the main points of the story.
 S4 The explanation explicates the meaning or message of the story.

Figure 4: Questionnaire used to evaluate all three conditions, presentation order of the three groups was randomized for each participant.

	Summary	Framing	Creativity
D	1.79 ± 0.70 ¹	1.69 ± 0.61 ¹	3.00 ± 0.66 ¹
F	3.86 ± 0.85 ²	3.22 ± 0.92 ²	3.18 ± 0.67 ¹
H	3.74 ± 0.82 ²	2.49 ± 0.86 ³	3.14 ± 0.66 ¹

Table 1: Survey results: perceived quality of text as summary and framing, and perceived creativity of generating system (mean ± std) for the three conditions. Superscripts indicate groups with statistically significant differences (at least at the $P \leq 0.0001$ level).

Results

An online survey platform was used to carry out the study. 36 participants were recruited from the University of Os-nabrück through e-mail and social media. Main experimental data collected for each participant were the individual item scores and the three optional free texts. Collected data further included demographic data, English language proficiency and the order in which conditions were presented.

For each subject the responses of each item-group were averaged, which resulted in three continuous values per condition. The final C , F and S scores for each condition were computed by averaging the condition's scores from all participants. The resulting ratings of the three conditions are reported in Table 1.

Evaluation

The gathered experimental data allows answering our research questions from the introduction by checking for significant effects of the factor 'condition' (D , H and F) on the three dependent variables: S , F and C . At the same time, it appears expedient to validate the experimental setup, by checking whether the factor 'presentation order' (the 6 possible permutations in which participants might have been presented with the conditions) has, as predicted, no effect on the dependent variables. Since a within-subject design was selected, this can be done by performing one two-factor repeated measure ANOVA for each of the three variables.

Since a Mauchly Test performed on all ratings showed

that the sphericity assumption is violated in the data, in the following, all ANOVA results are reported with a Greenhouse-Geisser correction.

Summary Quality

The null hypothesis regarding summary quality is that no differences exist between the quality of the three summaries.

	SS	df	MS	F	P value
Condition	96.95	1.96	49.34	100.53	2.81e-14
Order	7.41	9.82	0.75	1.54	0.16
Error	28.93	58.95	0.49		

Table 2: Greenhouse-Geisser corrected results of a two-factor ANOVA on the ratings for summary quality.

The ANOVA results in Table 2 show that 'condition' has a strongly significant effect on summary quality. A post-hoc, pairwise Tukey HSD test showed that the data-driven summary was rated significantly lower than the human or framing based summaries, whereas the latter two show no significant differences between each other (see Table 1 for the respective means). This means that the null hypothesis can be rejected, and, especially, that the framing based summary performs at human level. It is essential to put this result in the right context. The program did not summarize a natural language text at human level, where it first would have to extract and analyse the semantic content. Instead, it created a summary for a story it generated itself and for which it accordingly already possessed a computational representation of the ground truth. Also, the generated language of the summary is based on fairly rigid templates, so that any number of iterations would quickly dispel all humanoid pretensions. Notwithstanding these proper reservations, the observed performance is no trivial feat. The employed questionnaire included items regarding abstract understanding and teleology (meaning/message), which go beyond mere selective recounting.

The results also show that presentation order had no statistically significant effect on summary rating.

Framing Quality

To establish whether a better summary provides a better framing for a story, the null hypothesis can be formulated that the two conditions *H* and *F* show no significant difference in framing quality as compared to condition *D*. This is grounded on the previously established observation that *H* and *F* are the better summaries. Since *H* and *F* themselves display no significant difference in summary quality, no prediction is made about their relationship towards each other.

	SS	df	MS	F	P value
Condition	42.43	1.94	21.88	46.37	6.11e-9
Order	5.87	9.70	0.61	1.28	0.27
Error	27.45	58.18	0.47		

Table 3: Greenhouse-Geisser corrected results of a two-factor ANOVA on the ratings for framing quality.

The ANOVA results in Table 3 show that ‘condition’ has a strongly significant effect on framing quality. A post-hoc, pairwise Tukey HSD test showed that all three conditions differ significantly among each other, which allows the rejection of the null hypothesis (see Table 1 for the respective means). It is interesting to note that the two summaries that were rated equally (*H* and *F*) still seem to present a differing ‘frameability’. This implies that summary quality is not the only factor contributing to framing quality. The performed statistical analysis can not provide an answer to the question what these other factors might be. Here, the qualitative data collected using a free text field for each condition, asking if and how the presented text affected the participants’ perception of the story, can give further insights. Its analysis can be found at the end of this section, under the heading ‘Qualitative Evaluation’. For now it should suffice to say that we hypothesize that one such reason might be a phenomenological gap between summary *F* and readers’ mental models. Summary *H* mainly provides coverage of the setting and events happening in the story world (contentual level), whereas summary *F* also analyses the actions as standing in a functional context, e.g. withholding the fruits of the protagonist’s labour is described as a retaliation (artefactual level, perhaps even thematic if retaliation is taken to be the theme of the whole story). Following the assumption that thinking about a story in contentual rather than functional terms is more natural for laypeople untrained in the arts of narratological analysis, this would manifest in a phenomenological gap when reading condition *F* but not *H*. This should provide readers with a stronger impetus to re-contextualize the text, thus framing it ‘as’ something.

The ANOVA results again show that presentation order had no statistically significant effect on participants’ ratings.

Perceived Creativity

To establish whether a better framing leads to a higher perceived creativity of the generating algorithm the null hypothesis can be formulated that all three conditions show no significant difference in creativity ratings, since the three conditions all differ in framing quality.

	SS	df	MS	F	P value
Condition	0.60	1.71	0.35	2.93	0.09
Order	1.73	8.53	0.20	1.70	0.12
Error	6.10	51.20	0.12		

Table 4: Greenhouse-Geisser corrected results of a two-factor ANOVA on the ratings for perceived creativity.

First, it should be noted that the ANOVA results in Table 4 again show that presentation order had no statistically significant effect on participants’ ratings, which conclusively corroborates the choice of a within-subject design by demonstrating that subjects’ judgements were not biased by previously read conditions.

The results also show that ‘condition’ has no significant effect on perceived creativity, which means that the null hypothesis has to be accepted. This is unexpected since, as outlined in the introduction, the field operates under the assumption that perceived creativity should benefit from framing. One explanation that would allow to uphold this assumption might be what we would call the *weak framing assumption*, which would hold that systems do benefit from framing, however only in comparison to systems that perform no framing, while differences in framing quality do not propagate on creativity ratings (which would form part only of the *strong framing assumption*). This assumption remains unfazed by the present results, since no creativity ratings were solicited without framing. However, no reason comes to mind why framing quality should be irrelevant. It should also be observed that the quality of both, summary and framing, for condition *D* are consistently rated very low, and that it contains a logical non sequitur regarding the mechanics of bread-baking, which cast its adequacy as framing in a doubtful light—if accepted, such a perspective would hold that condition *D* was essentially unframed, which then would put even the weak framing assumption under pressure.

Another avenue at interpreting this outcome is by closer scrutinizing the numerical results. It is conspicuous that all three *C* values are located so close to the middle of the rating scale. Such behavior was recently also observed by Riegl and Veale (2018), who interpreted it as a symptom of participants’ boredom or overtaxation. Beyond questioning the data, only a closer look at the item-based breakdown of the question group ‘creativity’ remains (see Table 5). While this might aid in satisfying one’s curiosity, it should be clear that any analysis searching for significant difference on an item-based level remains prohibitive, because it would constitute a retesting of the same data, and be thus prone to false positives.

Considering the individual items C1 through C6 in Table 1 it becomes clear that a summary-based framing can not be expected to contribute equally to the individual ratings. The typicality (C2) and the novelty (C3) of a story are less likely to be significantly increased just by the merit of a short summary. On the other hand, a fitting but unexpected summary can well be taken as an indication for a

	interesting	typical	novel	imaginative	appreciative	skillfull
D	2.94	3.58	2.28	3.06	2.69	3.47
F	3.17	3.72	2.44	3.28	3.06	3.39
H	3.17	3.89	2.28	3.08	3.08	3.36

Table 5: Item-based breakdown of the question group ‘creativity’, reporting the condition-wide means for the items C1 through C6 (labeled with the quality they intend to capture for better readability).

system’s ability for appreciation of its own product (C5). Indeed, the appreciation scores seem to suggest an effect of the two ‘good-summary conditions’. As mentioned, a statistical test of this effect can unfortunately not be conducted. Thus, questions remain.

Qualitative Evaluation

The free text data collected from participants in order to understand how the different summaries might have affected their perception of the story can be used to analyse why conditions *F* and *H* differ in framing quality, while there are no significant differences in the respective summary qualities. A first corroboration of this statistical result in the introspective data is the fact that this voluntary field contained 12 relevant⁷ answers for condition *F*, while only 6 for *H*. This implies that subjects’ perception of the story was stronger affected by condition *F*, which is an indicator of more effective framing.

To analyse how subjects’ comments differ between the two conditions, two coders were employed to code all of the 18 comments. Their task was: “For each description, select one to three categories, which best describe what aspect of the reader’s perception was changed by, or at least was different in, the summary”. The categories available as codes were explained as follows:

- *function*: The text describes a change in perception of the function of certain events for the story as a whole. This includes the judgement that events form part of a high-level structural unit, like a ‘hero’s journey’, are fulfilling a narrative function like ‘introducing a conflict’, or take on an unexpected meaning like ‘deceiving an opponent’.
- *character*: The text describes a change in perception of characters or their interrelation. This can include individual’s motivations, emotions or reasoning, their perceived personality as well as attitudes towards each other.
- *theme*: The text describes a change in perception of the story’s moral (example moral: ‘don’t stray from the right path’), or which abstract themes of the human condition it represents (example: ‘search for the meaning of life’).
- *other*: everything that doesn’t fit the above categories.

Since any of the texts can contain commentary on several of these aspects, the results were interpreted as a one-to-many classification, for which recently a Cohen’s kappa-like measure of inter-coder agreement called Fuzzy Kappa was introduced (Kirilenko and Stepchenkova 2016). In our

⁷This count excludes answers like “It had no effect on my perception”, which were filtered out before all further analysis.

data, fuzzy unweighted kappa between the two coders is 0.60, which according to Landis and Koch (1977) is the border between moderate and substantial agreement. The aggregated results of the two codings of subjects’ comments can be found in Table 6, which depicts which percentage of codes were of which type in the two conditions. The most marked difference between condition *F* and *H* can be observed in the proportion of codes of the type ‘functional’. Both coders determined that the latter condition did not affect subjects’ perception of what certain events meant for the plot, while in the former condition around 30% of the indicators of differing meaning referred to this category. The other three codes do not yield such conclusive differences.

Our hypothesis, already outlined above, is that the functional perspective taken in condition *F* is uncommon to lay readers and for this reason works as a framing. However the analysis here is only a further indication for our case, and should be best read as a correlation: the summary that affected subjects’ perception of the story on a functional level happens to be the summary that is judged to be the better framing. To prove causation, more study would be required.

		character	function	theme	other
coder 1	F	0.56	0.31	0.13	0.00
	H	0.50	0.00	0.25	0.25
coder 2	F	0.53	0.27	0.13	0.07
	H	0.50	0.00	0.17	0.33

Table 6: Distribution of codes per coder and condition, as relative frequencies. Each row represents an overview of how often subjects’ perception of the story was affected in a specific domain (i.e. code) by the respective condition’s text.

Conclusion

The study presented in this paper has shown three things. First, it has demonstrated that an FU-Analysis based approach to story summarization can perform at human level— if employed on top of a plot generation system that implements the phenomena required to model functional units. By comparing the suitability of summaries of different quality for framing stories it, secondly, has shown that a better summary is also a better framing. This opens up an interesting avenue for storytelling systems to perform summary-based framing of generated artefacts, a further step up climbing the meta-mountain (Colton and Wiggins 2012). However, this is not the whole story, as summaries of comparable quality have shown differing suitability for framing. Our tenta-

tive advise to researchers interested in employing summary-based framing is to aim for creating a phenomenological gap between the level of abstraction at which consumers and the system reason about the plot. One possible approach to achieve this is FU-Analysis based summarization. More research is needed into how to generate summaries at other levels of abstraction, like for instance the thematic level concerned with higher meaning, symbolism or messages. Interestingly, the morals that Minstrel (Turner 1993) provided for its knight stories can be seen as one instance of framing based on thematic summary, developed long before the term framing itself was coined. Third, the study failed to show an effect of framing on the perceived creativity of a computational system. So far, it remains unclear to us whether this is due to the complex design of our study, an overly general creativity questionnaire or, perhaps, to the fact that framing actually doesn't work. Thus, we invite researchers with systems that practice different kinds of framing, or framing in a different domain than narrative, to explore whether they can replicate these results.

Acknowledgment.

We are grateful to Dr. Annette Hohenberger from University Osnabrück for her guidance in creating the questionnaire, and to Sven Wilke for the underlying implementation. This work was funded by an Alexander von Humboldt Ph.D. fellowship.

References

- Berov, L., and Kühnberger, K.-U. 2018. An evaluation of perceived personality in fictional characters generated by affective simulation. In *Proceedings of the Ninth International Conference on Computational Creativity*, 24–31. Salamanca, Spain: Association for Computational Creativity.
- Berov, L. 2017. Steering plot through personality and affect: an extended bdi model of fictional characters. In *KI 2017: Advances in Artificial Intelligence: Proceedings of the 40th Annual German Conference on AI*, 293–299. Cham: Springer.
- Charnley, J. W.; Pease, A.; and Colton, S. 2012. On the notion of framing in computational creativity. In *Proceedings of the 3rd International Conference on Computational Creativity*, 77–81.
- Colton, S., and Wiggins, G. A. 2012. Computational creativity: the final frontier? In *Proceedings of the 20th ECAI*, volume 12, 21–26. IOS Press.
- Colton, S.; Charnley, J. W.; and Pease, A. 2011. Computational creativity theory: The face and idea descriptive models. In *Proceedings of the 2nd International Conference on Computational Creativity*, 90–95.
- Colton, S. 2008. Creativity versus the perception of creativity in computational systems. In *AAAI Spring Symposium: Creative Intelligent Systems*, 14–20.
- Colton, S. 2012. Evolving a library of artistic scene descriptors. In *Evolutionary and Biologically Inspired Music, Sound, Art and Design*. Springer. 35–47.
- Eder, J. 2010. Understanding characters. *Projections* 4(1):16–40.
- Gambhir, M., and Gupta, V. 2017. Recent automatic text summarization techniques: a survey. *Artif. Intell. Rev.* 47(1):1–66.
- Goyal, A.; Riloff, E.; et al. 2013. A computational model for plot units. *Comput. Intell.* 29(3):466–488.
- Kirilenko, A. P., and Stepchenkova, S. 2016. Inter-coder agreement in one-to-many classification: fuzzy kappa. *PLoS One* 11(3):e0149787.
- Landis, J. R., and Koch, G. G. 1977. The measurement of observer agreement for categorical data. *Biometrics* 159–174.
- Lehnert, W. G. 1981. Plot units and narrative summarization. *Cogn. Sci.* 5(4):293–331.
- Norton, L. 1917. The richard mutt case. *The Blind Man* (Vol. 2):5.
- Rhodes, M. 1961. An analysis of creativity. *Phi Delta Kappan* 42(7):305–310.
- Riegl, S., and Veale, T. 2018. Live, die, evaluate, repeat: Do-over simulation in the generation of coherent episodic stories. In *Proceedings of the Ninth International Conference on Computational Creativity*, 80–87. Salamanca, Spain: Association for Computational Creativity.
- Ritchie, G. 2007. Some empirical criteria for attributing creativity to a computer program. *Minds Mach.* 17(1):67–99.
- See, A.; Liu, P. J.; and Manning, C. D. 2017. Get to the point: Summarization with pointer-generator networks.
- Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.
- Turner, S. R. 1993. *Minstrel: a computer model of creativity and storytelling*. Ph.D. Dissertation, University of California at Los Angeles, Los Angeles, CA.
- Wilke, S., and Berov, L. 2018. Functional unit analysis: Framing and aesthetics for computational storytelling. In *Proceedings of the 7th International Workshop on Computational Creativity, Concept Invention, and General Intelligence*, volume 2347 of *CEUR Workshop Proceedings*.

Critique as Creativity: Towards Developing Computational Commentators of Creative Works

Douglas H. Fisher

Computer Science
Vanderbilt University
Nashville, TN 37235

douglas.h.fisher@vanderbilt.edu

Haerin Shin

English; Media Studies

Vanderbilt University
Nashville, TN 37235

haerin.shin@vanderbilt.edu

Abstract

If beauty is in the eye of the beholder, then creativity can be sought in computational commentators on arts and artifacts across diverse form and media. While most research in computational creativity focuses on the generative nature of creativity, we address how interpretative aspects of creativity can be manifest by and implemented in computational agents as well.

Introduction

Human commentators, both professional and amateur, are acting creatively by making connections between their own perspectives and social context, with an artifact. Their interpretation, comparisons, and criticisms of artistic, literary, and engineering creations are colored by their experience and aesthetic. For example, a song that elicits a rich inner animation by the listener represents a highly personal remediation of the music and lyrics, and illustrates that creativity is both generative and interpretative.

This paper focuses on interpretative processes that yield narrative critique, criticism, and commentary, which we informally regard as synonymous. Stiny and Gips (1978) were early proponents of this (a reviewer pointed us to that book), though penetration of it into CC is limited, and a deeper comparison than we have made here is warranted.

Smith (1991) suggests that these interpretive artifacts lie on a continuum between commentaries on specific subject texts (or other artifacts) to works of secondary literature that go well beyond the subject work that inspired the commentary. To Smith's long list of commentary types we add (*peer*) review, which is at the specific-subject end of the continuum, and a good candidate as an initial computational commentator. Our position paper discusses computational commentary much more broadly than review, but we aspire to an implementation and return to the possibility of such a reviewer (e.g., of ICCS submissions).

Desiderata for Computational Critics

Several interacting capabilities seem desirable in a critic, computational or human, though these capabilities will vary with the types of commentary. *First*, computational critics should understand aspects of the medium-specific

traits and formal characteristics of the creation (Hayles 2004). Formal elements in literary expressions would include style (tone, diction, syntax, and structure), for instance; cinematography, mise-en-scene and montage in the case of film or other visual works; material texture in sculptures; compositional styles in music; clarity, sectioning, formatting in conference submissions; to name a few. Approaches in this vein include Russian formalism, New Criticism, and Barthes (1967) declaration of "The Death of the Author." Attention to formal aspects enables deeper understanding of the effects that a given creative product wields upon its viewers/readers. An understanding of medium-specific traits by an interpretative agent can also inform remediation of material between media.

Second, computational critics can use their personal and interpersonal understandings for finding connections to authorial intent in the creation, whether the critic reads the authorial intent "correctly" or not. Authorial intent can enrich the more immanent aspects of creative expression, as in biographical criticism (e.g. by Samuel Johnson, in *Lives of the Poets*) or the Romanticist vision of the creative genius (e.g. Wordsworth: a poem should be "the spontaneous overflow of powerful feelings" to be "recollect[ed] in tranquility" *From the Preface to the 2nd edition of Lyrical Ballads*, quoted Day, 2). A music video, for example, might enrich (or displace) a listener's internal remediation of a song; a conference review can suggest a related and productive direction for research.

Third, computational critics should be able to reason about the products (i.e., critiques) that they produce, and their relationship to the subject creation. One desirable characteristic of a critique is conceptual cohesiveness of the argument in the critique itself -- does the critique offer an interesting and informative interpretation of the subject artifact? Does a review adequately summarize a paper? If so, then at least in the "mind" of the agent writing the critique, the artifact is interpretable. Our concept of *interpretability* of artifacts is related to Bodily and Ventura's (2018) concept of *explainability*, which we elaborate later.

Fourth, a computational critic will, ideally, have the capacity to situate products within socio-historical contexts - namely engage in social criticism, perhaps with recommended citations in a review. Creative expressions not

only represent, but also critically reflect on and inspire reality; their contextual significance is therefore crucial to assessing the perspectival originality of a given work.

Fifth, a computational critic should be able to gauge how readers, viewers, and appreciators will react to the subject, thereby endowing paratextual value to the affective dynamics of a creative expression. Wolfgang Iser's "reader-response theory" (1978), Stanley Fish's (1970) "affective stylistics", David Bleich's emphasis on the subjective dimension of reader response (McCormick 1985), Norman Holland's (1989) focus on the psychological motivations that affect the reader's mode of engagement, and the attention Fish directed to the social and communal nature of reader response (Regis 1976), all fall under this category.

Critics Influence Creative Ecosystems

Several of these desirable capabilities highlight the social aspects of criticism. Humans are typically cognizant that other humans will critique their creative works, suggesting an ability to apply theory of mind when creating (Slater and Bremner, 2011). In the future, perhaps, computational creators will be cognizant that AIs (and humans) will comment as well. Computational critics might reasonably provide, at scale, what Ventura (2017) calls *external evaluation* -- critiques offer feedback that the subject's creator can use to learn or revise its own aesthetic(s). This possibility of computational critics acting, at scale, on human creators particularly, perhaps children, suggests Ethical Considerations too, which we discuss later.

Finally, criticism is a creative act in and of itself, for it requires an understanding of the myriad faculties of creativity, which must then be communicated in a persuasive manner, typically in natural language. Criticism has aspects of both interpolation and extrapolation, such as connecting the dots in a mystery and elaborating on the possible feelings of a minor character in a novel, respectively. The multivalence of criticism leads to benefits to the ICCC community of developing computational critics, perhaps the most obvious of which is that criticism represents a literary tradition that is under-represented in computational narrative generation, and in some human literary traditions as well (Smith, 1991).

Outline of the Paper

In the remainder of the paper we elaborate generative and interpretive perspectives; summarize prior work on evaluating creativity; forward cognitive architecture and ontology considerations for computational critic design, and address ethical considerations of computational critics. We conclude with a summary of main points, but a summary grounded in a prospective implementation of a computational reviewer of conference papers, notably ICCC, which we think is achievable as a proof-of-concept in one year.

From Generation to Interpretation

A common assumption is that human creative endeavors are produced from within a (self-)conscious liberal subject that designs one's creation. Intentionality, in this light, may

be understood as the vector of consciousness emanating out of a creator. We call this the Romanticist generative model of human creativity.

Computational creativity is indebted to the Romanticist vision of a creator with complex internality (Wang 2000), but the current state of AI is not at a level where phenomenological intentionality is the core propellant of creating agents. Indeed, many would question whether a computational agent has intent at all. Without intent, computationally created artifacts might be conceptually indistinguishable from natural objects that we deem to be beautiful, sublime, or otherwise evocative as Kant observes in *Critique of Judgment* (1773). Heidegger explains that a work of art (intentionally-created expressions) must reveal manifold meanings that survive the tests of time, giving rise to new values and interpretations across contexts, cultures, and temporalities (2002, pp. 1-56). Calling this function "worlding," Heidegger subtly identifies imaginative intent as a driving force behind creativity of human art.

Computational Intentionality & Authenticity

Computational creativity researchers do not currently require the high bar of consciousness for a claim of intentionality (Ventura, 2017), but only that the computational creator guides search by goals and/or utilities, for example. Another suggestion of intention is that an agent can "explain" why decisions were made in creating artifacts, perhaps by AI credit and blame assignment.

Another aspect of the internal processing of computational creators is the experience that the agent draws upon. For instance, Colton et. al. (2018) closely examines expectations for authenticity in a computational creation, drawing on Walter Benjamin's notion of the "aura" (2008), among others. While Colton et. al.'s description of the aura highlights the spatial and psychological "distance" between the artist/artwork and the appreciators as the source of a given work of art's ability to inspire affective impact, we would also like to call attention to Benjamin's point about how the "aura" gravitates toward the material progeny, characteristic singularity, and contextual originality with the advent of mechanically reproduced (e.g. Andy Warhol) or readymade art (e.g. Marcel Duchamp's Fountain).

One of Colton et. al.'s (2018) cautions is that discovery that an AI has no human experience may result in its rejection by humans, and that appropriate actions can be taken. In something of a contrast, Gunkel (2017) suggests that "... *some brands of aesthetic theory, like the various versions of formalism, will be more open to and accommodating of machine-generated content than others, like Romanticism and its veneration of the figure of artistic genius.*"

Seen in this light, the value of a computationally created product becomes subject to external reception, as well as inherent qualities of the work in question or the software developers of the agent that created the work.

External Interpretation of a Creation

While approaches that underscore the networked effect of the creative expression's external impact such as reader-

response, social, or formal criticism may be readily employed to assess the “aura” of a computationally creative agent, said aura cannot be attributed to the creator as an actor with conscious intent. Google’s Deep Dream is a case in point. The abstract visuals it produced were appreciated as a “technological novelty,” resulting in commercial success, but the metrics of their assessment were geared more towards this novelty itself than the meaning the visuals convey in comparison to other works that share stylistic traits, as would be the case with human-made abstract paintings. The Magical Realism Bot is another example, where the artistic effect of cognitive dissonance it employs largely banks on the readers’ marvel at the algorithm’s ability to emulate the stylistics of magical realism, as well as the aphorisms themselves. The novelty of a computer-made object then, ultimately falls under the eye of the beholder (i.e., p-creativity as described by Boden, 2004), which reverts agency back to human interpreters.

This leads to another point on the Romanticist generative model, and again related to responses of Colton et., al. (2018) when faced with unmasking of inauthentic agents. Jean Baudrillard maintained that representation could copy the original and become a reality unto its own in his conceptualization of the “hyperreal” (1994). The output of computational creativity could be deemed “hyperreal” in that they do not intentionally attempt to represent any existent modes of reality, but instead process data/artifacts to generate new variations. The originality of any “hyperreal” object or phenomenon is assessed not based on its formal qualities but more on its contextual significance; for instance, the Epcot World Showcase section in Disney World is deemed hyperreal because its simulation of world cultures belies its aspiration to authenticity by reflecting the stereotypes pertaining to cultures rather than their genuine reality as lived experience.

In sum, the generative model is productive from our perspective in that it pushes humans to reflect on the parameters of creativity and why it has been considered as uniquely human. But creativity is not simply centrifugal -- creativity is social and multivalent too, dependent upon the multilateral dynamics of the social context across times and cultures, there is the need for an interpretive agency, which can be computationally modeled. This is why we propose the interpretive model as an addition to the (Romanticist) generative model, asserting that interpretation is an exercise of imagination. An act that bridges the subjective and the objective, critique demonstrates that the two modes of agency (generative and interpretive) are co-constitutive rather than binary oppositions.

Assessing Creativity

Our expectation is that computational critics (and criticism) will be judged by their (its) conversancy about and synthesis of diverse works, creators, aesthetics, genre, and other critics. While understudied, we believe that computational criticism and critics will build on prior work on characterizing and assessing human and computational creativity. For example, computational creators internally

assess their intermediate states in the act of creating, as well as assessing the outputs of creation. The measures and mechanisms used to make these assessments are undoubtedly related to, if not identical to some of the assessments that will be made by computational critics. Abilities to reason about criteria in this section have implications for Desiderata for Computational Critics discussed earlier.

This section focuses on prior work for assessing “artifacts as manifestations of creativity” -- many refer to these as “creative artifacts” and we will use that as shorthand too. In what follows we focus here on criteria as applied to the artifacts that will be critiqued, but in all cases, different instantiations of these broad classes of criteria also can be applied to the artifact that the critic creates – a critique.

After criteria applied to artifacts, we touch upon characterizations and assessments of creative processes. Most prior work in assessing creativity assumes implicitly or explicitly a generative perspective, and so we will also comment on how some of the criteria for creativity might be adapted to an interpretive perspective and a computational critic more specifically. For example, to repeat the opening sentence of this section, critiques, which are the creative artifacts of critics, are often valued for their recognition of the diversity of perspectival values, where diversity of perspective and aesthetic may not be as relevant when discussing generative creative agents.

Novelty

Novelty refers to an overall assessment, often as a metric, of the differences and similarities between an artifact and others of the same kind. The generalization of “same” to “comparable” kind, to include some that span media and form, stems uniquely from an interpretive perspective. For instance, a written work can certainly be compared to other writings of the same author and genre in terms of their techniques and narrative content, but additionally, comparisons can be made to audiovisual media (e.g., most obviously a film adaptation of a novel). Indeed, comparability can be as broad as remediation between media allows (e.g., a bible story of Daniel in the Lion’s Den as a painting at the National Galleries). Colton (2010) also imagines very broad possibilities, based on shared ontologies and other resources, which we return to later.

Novelty can be judged relative to all comparable artifacts in the world, or it can be individualized to the artifacts experienced by an agent. The former is akin to Boden’s (1992) definition of historic (h-)creativity (where ‘creativity’ is limited to the novelty dimension), and the latter is akin to Boden’s (1992) notion of psychological (p-)creativity (novelty). An interpretive perspective suggests we generalize to a continuum that bridges small to large groups of persons. In sum, computational critics and their critiques will presumably be judged by the breadth of their knowledge base of artifacts, their comparable kinds, and their exposure to agent populations of varying scope.

Value

Artifacts can be novel but ineffective for utilitarian purposes or in eliciting an affective response. We identify four categories of value, which would be variously used by interpretive agents, including critics of art, film, book, engineering products, and humanistic works.

Exchange value corresponds to monetary valuations. Creative works (as in the domain of art) can be assigned exchange value when treated as commodity, in which case aesthetic appeal may be converted into monetary values (utility). The professionalization of art has boosted this tendency. For example Google's Deep Dream was sold for an exorbitant price at auction (Business Insider 2016).

Value can also refer to a **utilitarian value**. Maher and Fisher (2012) suggest that just as descriptive intrinsic attributes could be used to identify novelty, utility or functional attributes like processing speed or recyclability for laptops could be used to create a utilitarian space.

Value can be a way of measuring **technical sophistication**, reflecting the "craft" aspect of material manifestations of creativity. The value of a written prose piece, for instance, can be judged based on its skillful navigation of syntax, diction, structure, narrative feasibility, and other aspects that make the prose more appealing, logical, enjoyable, and rhetorically more effective. Technical sophistication would be used to partially assess the value of critiques themselves. In visual art, the creator's ability to effectively employ color combinations, apply proper strokes or perspectivalization can determine the quality of the product.

Lastly, value can be seen in terms of the **affective force** that a creative work evokes. While the above forms of value are always subject to social contextualization and subjective judgment too, this type of value is more pronouncedly non-axiological; namely, it is less likely to be mapped on to differential hierarchies that determine whether a certain type of object/phenomena is superior or inferior to others. Value can simply inhere in instances where a creative output could emotionally "move" the recipient.

Unexpectedness

Another common characteristic for judging creativity is unexpectedness or surprise. Grace and Maher (2014) have a good taxonomy and survey of unexpectedness. We call out two works here: novelty in a future space of anticipated (projected) artifacts (Maher and Fisher, 2012), and changes to the posterior (post-artifact) distributions (e.g., Baldi & Itti, 2010) and/or to conceptual structures (Grace, et. al., 2015), over a space of artifacts. Our reason for calling out these interpretations of unexpectedness is that they make historical context explicit, through a "weighting" of the past, even "curve fitting", for an assessment of the current artifact. The historical perspective suggests another characteristic which we would want critics to assess -- authority.

Authority and authority

By authority we mean a dimension that is akin to its use in social/citation networks (Kleinberg, 1999), with inward and outward pointing links to each node. Nodes with a

higher fan-in, for example, suggest greater authority. In the context of computational creativity, an artifact's history within a social network is reflected in the links between artifacts. Every time a variation on (aka derivation of) a creative artifact is made, its authority is increased. This perspective suggests that it cannot be evaluated until sometime after an artifact's introduction, because variations and interpretations of it must be made, and this may take time.

However, not only will the valued critic be able to trace and evaluate the descendants (in links) of an artifact, but a critic may be able to judge the capacity of an artifact to be reinterpreted and varied. Potential authority refers to the capacity for imaginative latitude in adapting an artifact. This category aligns with the principles of reader-response theory, but also highlights a creative artifact's inherent qualities that encode a greater degree of interpretive freedom. For instance, works that are deemed "canonical" retain their appeal across temporal and cultural transitions because they address issues that resonate with appreciators in a universal manner. In the case of Shakespeare, most notably, his ability to capture the most pressing concerns of the human society and heart has enabled his works to be reinterpreted, generating new value and appreciation centuries after their original debut. Impressionist paintings still evoke marvel in the viewers' eyes by accentuating the central role of light and their interplay with human perception. In sum, potential authority can be measured by an artifact's capacity, as determined through intrinsic and social factors, to command universality and generate new values. The specifics of such a measure remain a challenge.

Sosa and Gero (2005) use the term "authority" to mean hierarchical authority among agents/persons within a field of endeavor. More generally, they say "... *in agent societies with strong social ties uneven hierarchies generate powerful opinion leaders that exert the role of gatekeepers to the domain. In contrast, in social networks with weak ties, influence is distributed among adopters and the expert judgments tend to vary over time. Consistent with Gardner's (1994) observation, the former social arrangement generates higher variance in the distribution of prominence whilst the latter yields more egalitarian distributions.*" (Sosa and Gero, 2005, p. 26)

This is a different, but a related kind of authority, -- authorities of persons, not artifacts. It is yet another kind of authority, an awareness of which a critic should be assessed on, even if the critic adjusts for or ignores it in the critique. Colton (2010) made a similar point.

Like many conferences ICC 2019 has moved to blind review because it "*encourages submissions from authors in adjacent research fields.*" Though the call stops short of stating the rationale of this brand new policy as one of mitigating the influence, through review, of power hierarchies by opening gates to those outside the community, that desire to mitigate and expand is a natural inference.

Authenticity

We have already talked at length about Colton et. al.s (2018) concept of authenticity, and this too can be used by

critics, particularly when they know the creator and its capacity, if an AI, for empathy and other feelings. We return to authenticity under Ethical Considerations.

Explainability and Interpretability

Our conceptualization of interpretability is related to what Bodily and Ventura (2018) conceptualize as explainability, but rather than being an internal capacity of a creative agent, explainability can also happen external to the creator, in another agent, as part of an interpretive capacity that we call interpretability.

Bodily and Ventura give broad examples of explainability by a creative agent, to include stating its “feelings” and goals in creating an object, as well as the processes (“logical rules”) used in the creation. Specifically, explainability is the capacity of an agent to explain why its creation is consistent with the agent’s aesthetic values. Our second-to-last value type of technical sophistication is particularly related to the concept of explainability, but other types can probably be cast here too (e.g., “why did that computational painting bring so much at auction?”)

By interpretability we mean explainability (by the critic, not the creator) using any (or a selected) set of aesthetic values that are at the disposal of the critic. Because the critic is not generating the artifact, “proving” interpretability by demonstration -- that is, finding an interpretation -- has non-deterministic aspects, that can be addressed by searching a knowledge base of past products and aesthetic rules for those that fall within an aesthetic class, as well as using other information like authorship, location of creation, and other metadata to hone in on relevant concepts using established ontologies known by the critic. AI argumentation (Bench-Capon and Dunne, 2007) will be important in the writing of critiques that stand up to scrutiny, including lawsuits, as well as demonstrating to humans what a well-supported critique looks like, which has Ethical Considerations as well.

Social Interactions

We have already addressed social interactions at length, with information like authorship; authority as citation and influence within a community; and a vision for an ecosystem of creators and commentators.

Assessing Creative Processes

Up to now we have addressed creativity manifest in artifacts, as well as social processes that surround them. The processes that create artifacts internal to the creator can also be part of the critiques, if they are known. Stokes (2005) suggests, using an AI search framework, that an

ideally creative process is not too under-constrained (e.g., without an aesthetic guide) and not too over-constrained (e.g., a near straight line to a goal, as in a mediocre “new” car design). This dichotomy is related to the exploration and exploitation tradeoff, which is ubiquitous in AI, and with continuous space for characterizing creativity between unconstrained exploration and rigid exploitation.

Stokes asserts that any creativity starts with an ill-defined problem space, which the creative agent can better define. It is this defining process where much of the agent’s creativity is found, and the ability to better define a problem space is yet another possible definition of intentionality. For example, “*Monet precluded dark-light contrasts*” (Stokes, 2005, p. xiiv) and Jimmy Breslin sought out interviews that no one else thought to do (e.g., Breslin, 1963). These definitional choices help to ensure the novelty, unexpectedness, perhaps value of the final artifact, long before it is finished. Novelty and value can be applied to the earliest internal states of the creation process. Harkening back to Baudrillard and Heidegger, the idea of “originality” is always contextual. Originality can be, for instance, referring to whether the creating agent was able to produce the output mainly based on one’s own imagination without starting the work using a similar object as a seed, though this too illustrates a binary that can be sublated.

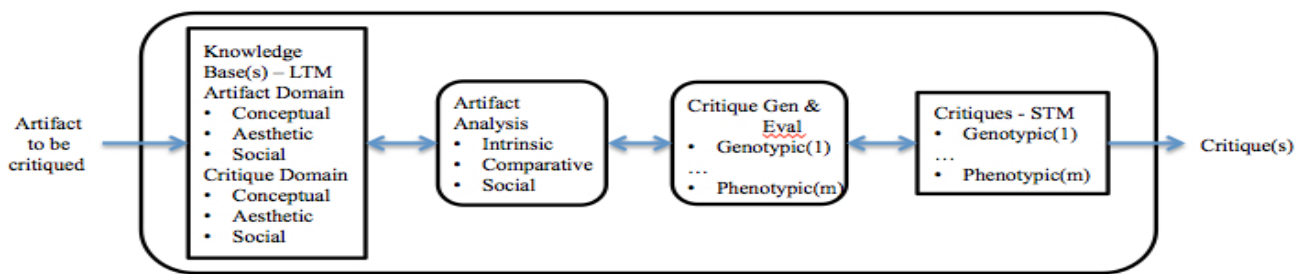
While a computational critic will need generative processes in producing its critiques, and will use much of the technology developed for other creators of narrative (e.g., Riedl and Bulitko, 2013), its ability to peer into “thought” processes of creators will be limited, so this sophisticated theory of mind (Prabowo & Thelwall, 2013) would not be a near-term focus, but we pose it as a challenge.

Architectures and Ontologies

Critics should have a capacity to summarize an artifact; to compare it to other artifacts for purposes of assessing its apparent creativity; to reason about its creator, where possible; to place the artifact in social and cultural context; and to organize and write an interesting and “fair” critique.

Figure 1 brings together our various discussions by illustrating a general architecture for a computational critic. It profitably compares and contrasts with Stiny and Gips (1978, Ch 2), inviting a deeper analysis and synthesis later.

An artifact to be critiqued is presented to the system of Figure 1, and undergoes analysis of its intrinsic properties and metadata, as well as comparisons to other works and abstractions in a knowledge base to assess novelty, value, unexpectedness, authority, interpretability relative to aesthetics at its disposal, and other characterizations.



The critic will perform these assessments by consulting conceptual, aesthetic, and social knowledge bases. For example, assessments will typically compare an artifact to a set of artifacts based on a set of intrinsic relationships (e.g., the ratio of the length of a car’s hood to the total length of the car, in an engineering example), which are often unary, also called attributes (e.g., the color of a car). If artifacts are represented by attributes only, then this representation is typically referred to as a feature vector. If object representations include one or more greater-than-unary relationships, then these are relational representations, which are naturally visualized as graphs. Triples representations represent all knowledge in binary relations between objects; such uniformity is desirable in many settings. The relationships used to do comparisons can be the union of relationships ever used for instances of a kind of artifact, or they can be individualized, both with regard to what relationships are included in comparisons and in how these relationships are weighted for producing an overall assessment. Extreme points in knowledge representation create another continuum on which critics and critiques will fall.

The particular relations in the knowledge bases would be based on ontologies that could be hand crafted and/or built through processes of conceptual blending and analogical reasoning (Colton, 2010). There are other ontologies that cover diverse topics that would be good starting points (e.g., [BBC’s Ontologies](#)).

Endowing computational critics with knowledge, to include of other agents, their works, and relationships between them, will likely be one distinguishing characteristic in the design and behavior of computational critics from computational generators of artifacts. So far, this latter group is not concerned with other creators, nor diverse aesthetics, compared to their human creator counterparts.

The architecture of Figure 1 is informed by Ventura’s (2017) architecture for a (generative) computational creativity system. In particular, Ventura proposes an architecture for CC systems that differentiates “genotype” and “phenotype” representations and evaluations. A genotype of a creative artifact is a private precursor to a public phenotypic representation. Instead of one, we imagine that there will be a series of genotypic representations, starting with a concept map (Nowak & Cañas, 2006), with subsequent genotypes adding annotations from the various knowledge bases outlined, then following revision steps similar to narrative generation found in Callaway and Lester (2002), for example, including a narrative planner (i.e., narrative organizer, sentence planner, revision component, and surface realizer). This planning and writing

apparatus, together with creativity pointing evaluation functions at each step of genotype and phenotype, would fill the Gen&Eval module of Figure 1.

Other architectures include Romero et. al. (2003), which used computational critics for rating candidates during an evolutionary creative algorithm. They also have a vision of a society of creators and commentators.

Ethical Considerations

There are many ethical concerns that computational critique implicates, to include so called fake news and memes, both of which are in easy reach of generative computational creativity; modeling civility and productivity in commentary, even generosity (Coman, et. al., 2018); the obligations and conventions of citing AI commentators; the legal responsibility and liability of AI commentaries; the implications of discovered inauthenticity to certain persons, like children; to name but a few. We save most of these for another day, and concentrate on two concerns that follow from the (Romanticist) generative perspective.

Implications of the Generative Perspective

While persuasive in its appeal to highlight the subjectivity of the creator, the generative perspective carries the risk of reinforcing anthropocentrism when applied to computational creativity. Under its rubric, the AI either acquires a metaphorical equivalence to the genius figure who possesses a singular talent to be appreciated, or is dismissed as an instrumental actuator of human will, denigrated to the level of a mere tool. Either way, the exaltation of the creator becomes a celebration of what we deem to be essentially human. This process, in turn, re-fetters humanity to a hierarchical ontology that has been plaguing us for eons, for the gist of the undergirding logic here is that those who possess special talent or skills (and subsequently their creations) are “better” than others. Attributes that are commonly attached to creativity, such as novelty or authenticity, demonstrate the danger of this discursive framework. These characteristics also apply to computational critics, because they will generate artifacts as well.

Commentary at Scale of Human Creations

What will an ability to computationally critique works imply – particularly critiquing human creative works and particularly when done at scale? Even if computational commentators are intended to critique computational generative agents, there seems no reason that these commentators could not be applied at scale to human works, and not

just to professional human creators, but all groups, including children. Computational commenters, particularly mediocre ones that gain from anthropomorphism, could squelch creativity rather than beneficially add to creative ecosystems. There is already debate on the pros and cons of automated essay graders on the mindset of students. (Smith, 2018). Granted, robo-graders are rudimentary commentators at best, but their effects can be significant. Our intent here is that computational creativity research raises the level of commentary, and indeed, the best computational commentators may model what many agree is noble behavior by removing or managing cultural bias. We take up this latter theme in the next section.

Implications of Adherence to Authenticity

Authenticity presupposes the existence of a grounded truth that deserves to be respected. High-fidelity to the “real” can render alternative modes of representation and perception marginal or even inferior, as we have seen in Plato’s Allegory of the Cave (2013). Platonic idealism, which suggests that there exists a transcendental “truth” while their material instantiations are ephemeral and therefore inferior “shadows,” is in essence a precursor of Rene Descartes’ substance-dualist affirmation of humanity’s superiority over other forms of being, as he had asserted in “Animals are Machines” (Harrison, 1992).

Another detrimental form of adherence to “authenticity” can be seen in the domain of ethnic literature. Racial and ethnic minority writers are often defined mainly by their hyphenated identity (e.g. Asian-American, African-American), and are therefore expected to serve as metonyms of their identificatory peers. A case in point is the unfavorable response that Korean-American writer Chang-rae Lee encountered upon publishing his third novel *Aloft* (2004) in which the protagonist is a white Italian American man. Having received critical acclaim by exploring the trope of being torn between two cultures or the stereotype of the “forever foreigner” in his debut novel *Native Speaker* (1995), and the traumatic legacies of coloniality across Japan and Korea from the perspective of an immigrant in his second novel *A Gesture Life* (1999), Lee directly countered the general readership and critics’ expectation that narratives pertaining to his own identity as an Asian immigrant are the only ones that carry authentic value. An ironic development, given how the predominantly white male writer-base in the cultural industry within the U.S. has been granted the liberty to assume a wide variety of identities in their fictional imaginaries, which attests to the “invisibility of whiteness” (Reddy, 1998).

But computational criticism can be an “equalizer” along dimensions of bias. This approach opens up new understandings of creativity, and art, in tow, granting agency to the medium itself in new ways. The concept of medium agency is already acknowledged by media studies (that a thing or phenomena can have agency or in other words do things and make things happen) without necessarily being a subject. N. Katherine Hayles’s (2017) exploration of nonhuman cognition, in this regard, is relevant.

Concluding Remarks and Future Work

This position paper has argued broadly for the importance of considering computational interpreters as part of the computational creativity landscape. We have summarized prior work on evaluating creativity in a generative system, with attention to how the interpretive perspective can add to this literature; forwarded cognitive architecture and ontology considerations for computational critic design; and addressed ethical considerations of computational critics. In the future near term, a deeper comparison with Stiny and Gips (1978) is needed. We also aspire to implement a computational critic in the near term.

On opening we suggested that a peer review of a conference paper submission would be a good first implementation of a computational critic. A review is grounded by a specific paper, in a specific context, which constrains the analysis immensely. Of the five desirable capabilities of the Introduction an ideal review instantiates all of these, with comments on scientific clarity and organization; suggestions for related research; self checking that the review is on point and helpful; suggestions of related work and otherwise appropriate citations; and conveying other helpful suggestions in the review that improves the paper. An ideal conference reviewer would also be aware of the many specific criteria of Assessing Creativity, like novelty and value, and would comment on and/or score these.

Conference submission reviews, which are text, also critique text, which means that a number of computational tools are available, like source-text summarization, related-work search, and topic modeling are available for a proof of concept. All in all, a proof of concept of a computational reviewer for a particular conference like ICCG seems achievable in one year.

Acknowledgements

This work was supported in part by NSF #1521672 “Collaborative Research: CompSustNet: Expanding the Horizons of Computational Sustainability”. Special thanks to the students in the Vanderbilt University Course on the Ethics of Artificial Intelligence, and to three anonymous reviewers for their very helpful comments, including pointers to Stiny and Gips (1978) and Stiny (2015); “owning” the newness of the interpretive perspective; ambitions for implementation and agenda; and importance of connecting concepts across the paper.

References

- Baldi, P. & Itti, L. (2010) “*Of Bits and Wows: A Bayesian Theory of Surprise with Applications to Attention*” *Neural Networks*. V.23, N.5: 649–666.
- Barthes, R. (1967) “*The Death of the Author*” in *Image, Music, Text*: 143-148. Trans. Stephen Heath. Fontana.
- Baudrillard, J. (1994) *Simulacra and Simulation*. The University of Michigan Press.
- Bench-Capon, T.J.M & Dunne, P. (2007) “*Argumentation in artificial intelligence*” *Artificial Intelligence*, V.171

- Benjamin, W. (2008) *The Work of Art in the Age of Its Technological Reproducibility, and Other Writings on Media*. The Belknap Press of Harvard Univ. Press.
- Boden, M. (1992) *The Creative Mind*. London: Abacus.
- Bodily, P. M., & Ventura, D. (2018) “*Explainability: An Aesthetic for Aesthetics in Computational Creative Systems*” International Conference on Computational Creativity: 153-160.
- Breslin, J. (1963). Digging JFK grave was his honor. The New York Herald Tribune, Opinion, November 1963.
- Callaway, C. B., & Lester, J.C. (2002) “*Narrative Prose Generation*” *Artificial Intelligence*, V.139,N.2: 213–52.
- Colton, S. (2010) “*Towards ontology use, re-use and abuse in a computational creativity collective*” In Kutz, O., Hois, J., Bao, J., Grau, B.C. (eds.) *Modular Ontologies – Proceedings of the Fourth International Workshop (WoMO’2010)*: 1–4, IOS Press.
- Colton, Pease, A., Saunders, R. (2018) “*Issues of Authenticity in Autonomously Creative Systems*” International Conference on Computational Creativity: 272-279
- Coman, A., Mueller, E.T., Mayer, M. (2018). International Conference on Computational Creativity.
- Fish, S. (1970) "Literature in the Reader: Affective Stylistics"
- Grace, K. & Maher, M.L. (2014) “*What to expect when you’re expecting: The role of unexpectedness in computationally evaluating creativity*” International Conference on Computational Creativity.
- Grace, K. and Maher, M. L., Fisher, D. & Brady, K. (2015) “*Data-intensive evaluation of design creativity using novelty, value, and surprise*” *International Journal of Design Creativity and Innovation*, V.3, N.4: 125-147
- Gunkel, D. (2017) “*Rethinking Art and Aesthetics in the Age of Creative Machines*” *Philosophy & Technology*, V. 30, N. 3: 263-265.
- Harrison, P. (1992) “*Descartes on Animals*” *The Philosophical Quarterly*, V.42, N.2: 219-227.
- Hayles, N. K. (2004) “*Print Is Flat, Code Is Deep: The Importance of Media-Specific Analysis*” *Poetics Today*, V.25,N.1: 67-90.
- Hayles, N. K. (2017) *Unthought : The Power of the Cognitive Nonconscious*. The University of Chicago Press.
- Heidegger, M. (2002) “*The Origin of the Work of Art*” in *Off the Beaten Track*, Cambridge University Press.
- Holland, N. N. (1989) *Poems in Persons: an Introduction to the Psychoanalysis of Literature*. Columbia Univ Press.
- Iser, W. (1978). *The Implied Reader*. Johns Hopkins Univ Press.
- Kant, I. (1973) *The Critique of Judgement*. Oxford, Clarendon Press.
- Kleinberg, J. (1999). *Authoritative sources in a hyperlinked environment*. *Journal of the ACM*, V.46, N.5. 604-632
- Lyu, S., Rockmore, D. & Farid, H. (2004) A digital technique for art authentication. *Proceedings of the National Academy of Sciences*, V.101, N.49:17006–17010.
- Maher, M.L. & Fisher, D. (2012) *Using AI to Evaluate Creative Designs*. ICDC 2012 - 2nd International Conference on Design Creativity.
- McCormick, K. (1985) “*Theory in the Reader: Bleich, Holland, and Beyond*” *College English*,V.47,N.8: 836-850.
- Novak, J. D. & Cañas, A. J., (2006) “*The Theory Underlying Concept Maps and How to Construct and Use Them*” Technical Report IHMC CmapTools 2006-01 Rev 2008-01 Institute for Human and Machine Cognition, Pensacola FL.
- Plato (2013) *Republic*. Harvard University Press.
- Prabowo, Rudy & Thelwall, Mike. (2013). Sentiment analysis: A combined approach. *Journal of Informetrics*. 143-157. 10.1016/j.joi.2009.01.003.
- Reddy, T. M. (1997) “*Invisibility/Hypervisibility: The Paradox of Normative Whiteness Transformations*” *The Journal of Inclusive Scholarship and Pedagogy*, V.9,N.2:55-64.
- Regis, Edward (1976) “*Literature by the Reader: The Affective Theory of Stanley Fish*” *College English* 38
- Riedl, M. & Bulitko, V. (2013) “*Interactive Narrative: An Intelligent Systems Approach*” *AI Magazine*. V. 34, N. 1.
- Romero, P., Machado, A., Santos, A., & Cardoso, J. (2003) “*On the Development of Critics in Evolutionary Computation Artists*” *Evoworkshops*.
- Slater, A. & Bremner, J. G. (2011) “*An Introduction to Developmental Psychology*” *British Psychological Society*, Blackwell Publishers.
- Smith, B. (1991) “*Textual Deference*” *American Philosophical Quarterly*, V.28, N.1: 1-13.
- Smith, T. (2018). More States Opting to ‘Robo-Grade’ Student Essays by Computer, NPR, aired June 30, 2018.
- Sosa, R. and Gero, J. (2005) “*A Computational Study of Creativity in Design: The Role of Society*” *Artificial Intelligence for Engineering Design Analysis and Manufacturing*, V.19, N.4: 229-244
- Stiny, G., & Gips, J. (1978). *Algorithmic aesthetics : computer models for criticism and design in the arts*. Berkeley: University of California Press.
- Stiny, G. (2015). The critic as artist: Oscar Wilde’s prolegomena to shape grammars. *Nexus Network Journal*, V. 17,N.3: 723-758
- Stokes, P. (2006) *Creativity from Constraints*. Springer Publishing Company, New York
- Ventura, D. (2017). “*How to Build a CC System*” International Conference on Computational Creativity.
- Wang, O. N. C. (2000) “*Kant's Strange Light: Romanticism, Periodicity, and the Catachresis of Genius*” *Diacritics*, V. 30, N.4: 15-37.

Combinets: Creativity via Recombination of Neural Networks

Matthew Guzdial and Mark Riedl

School of Interactive Computing
Georgia Institute of Technology
Atlanta, GA 30332 USA
mguzdial3@gatech.edu and riedl@cc.gatech.edu

Abstract

One of the defining characteristics of human creativity is the ability to make conceptual leaps, creating something surprising from existing knowledge. In comparison, deep neural networks often struggle to handle cases outside of their training data, which is especially problematic for problems with limited training data. Approaches exist to transfer knowledge from models trained on one problem with sufficient data to new problems with insufficient data, but they tend to require additional training or a domain-specific method of transfer. We present conceptual expansion, a general approach for reusing existing trained models to derive new models without backpropagation. We evaluate our approach on few-shot variations of two tasks: image classification and image generation, and outperform standard transfer learning approaches.

Introduction

Modern deep learning systems perform well with large amounts of training data on known classes but often struggle otherwise. This is a general issue given the invention or discovery of novel classes, rare or illusive classes, or the imagining of fantastical classes. For example, if a new traffic sign were invented tomorrow it would have a severe, negative impact on autonomous driving efforts until enough training examples were collected.

Deep learning success has depended more on the size of datasets than on the strength of algorithms (Pereira, Norvig, and Halevy 2009). A significant amount of training data for many classes exists. But there are also many novel, rare, or fantastical classes with insufficient data that can be understood as derivations or combinations of existing classes. For example, consider a pegasus, a fantastical creature that appears to be a horse with wings, and therefore can be thought of as a combination of a horse and a bird. If we suddenly discovered a pegasus and only had a few pictures, we couldn't train a typical neural network classifier to recognize a pegasus as a new class nor a generative adversarial network to create new pegasus images. However we might be able to approximate both models given existing models trained on horse and bird data.

Various approaches exist that reuse knowledge from models trained on large datasets for a particular problem to try

to solve problems with smaller datasets, such as zero-shot and transfer learning. In these approaches, knowledge from a source model trained is applied to a target problem by either retraining the network on the target dataset (Levy and Markovitch 2012) or leveraging sufficiently general or authored features to represent new classes (Xian, Schiele, and Akata 2017). The latter of these two approaches is not guaranteed to perform well depending on source and target problems, and the former of these is limited in terms of what final target models can be learned.

Combinational creativity is the type of creativity humans employ when combining existing knowledge to create something new (Boden 2004). Many algorithms exist that attempt to reflect this process, but they have historically required hand-authored graphical representations of input concepts with combination only occurring across symbolic values (Fauconnier 2001). A neural network is a large, complex graph of numeric values derived from data. If combinational creativity techniques could be applied to recombine trained neural networks, this could allow us to address the novel class problem (e.g. pegasus) without the introduction of outside knowledge or heuristics.

We introduce a novel approach, *conceptual expansion*, that allows for the recombination of an arbitrary number of learned models into a final model without additional training. In the domains of image recognition and image generation we demonstrate how recombination via conceptual expansion outperforms standard transfer learning approaches for fixed neural network architectures. The remainder of the paper is organized as follows: first we discuss related work and differentiate this technique from similar approaches for few-shot problems. Second, we discuss conceptual expansions in detail and the search-based approach we employ to construct them in this paper. Third, we present a variety of experiments to demonstrate the limitations and advantages of the approach.

Related Work

Computational Creativity

Combinational creativity represents both a type of creativity and class of algorithm for knowledge reuse through recombining existing knowledge and concepts for the purposes of inventing novel concepts (Boden 2004). There have many

prior combinational creativity algorithms. Case-based reasoning (CBR) represents a general AI problem solving approach that relies on the storage, retrieval, and adaptation of existing solutions (De Mantaras et al. 2005). The adaptation function has led to a large class of combinational creativity algorithms (Wilke and Bergmann 1998; Fox and Clarke 2009; Manzano, Ontañón, and Plaza 2011). These tend to be domain-dependent, for example for the problem of text generation or tool creation (Hervás and Gervás 2006; Sizov, Öztürk, and Aamodt 2015).

The area of belief revision, modeling how beliefs change, includes a function to merge prior existing beliefs with new beliefs (Cojan and Lieber 2009; Konieczny and Pérez 2011; Fox and Clarke 2009). Amalgams are an extension of this belief merging process that looks to output the simplest combination (Ontañón and Plaza 2010). The mathematical notion of convolution has been applied to blend weights between two neural nets in work that parallels our desire to combine combinational creativity and machine learning, but with inconclusive results (Thagard and Stewart 2011).

Conceptual blending is perhaps the most popular combinational creativity technique, though it has traditionally been limited to hand-authored input (Fauconnier 2001). Li et al. (2012) introduced goals to conceptual blending, which parallels our usage of training data to optimize the structure of a combination. Conceptual blending has further traditionally relied on symbolic values, which makes it ill-suited to statistical machine-learning. Visual blending (Cunha et al. 2017), combines pieces of images using conceptual blending and parallels our use of combinational creativity with Generative Adversarial Networks, however it requires hand-defined components and combines images instead of models. Guzdial and Riedl (2016) utilized conceptual blending to recombine machine-learned models of video game level design by treating all numbers as ordinal values, but their approach does not generalize to neural networks.

Combinational creativity algorithms tend to have many possible valid outputs. This is typically viewed as undesirable, with general heuristics or constraints designed to pick a single correct combination from this set (Fauconnier 2001; Ontañón and Plaza 2010). This limits the potential output of these approaches, we instead employ a domain-specific heuristic criterion to explore the space of possible combinations for an optimal one.

In Boden's model of three types of creativity (Boden 2009), we can consider our approach to combine elements of combinational and exploratory creativity. Conceptual expansion is a combinational creativity algorithm as, given a set of existing knowledge, it defines a space of possible valid combinations. We then employ a search process, which we call conceptual expansion search, to explore this space for particular combinations that meet some goal or heuristic.

Knowledge Reuse in Neural Networks

A wide range of prior approaches exist for the reuse or transfer of knowledge in neural networks, such as zero-shot, one-shot, and few-shot learning (Xian, Schiele, and Akata 2017; Fei-Fei, Fergus, and Perona 2006), domain adaptation (Daumé III 2009), and transfer learning (Lampert, Nick-

isch, and Harmeling 2009; Wang and Hebert 2016). These approaches require an additional set of features for transfer, or depend upon backpropagation to refine learned features from some source domain to a target domain. In the former case these additional transfer features can be hand-authored (Lampert, Nickisch, and Harmeling 2009; Kulis, Saenko, and Darrell 2011; Ganin et al. 2016) or learned (Norouzi et al. 2013; Mensink, Gavves, and Snoek 2014; Ba et al. 2015; Elhoseiny et al. 2017). In the case requiring additional training these approaches can freeze all weights of a network aside from a final classification layer or can tune all the weights of the network with standard training approaches (Wong and Gales 2016; Li et al. 2017). As an alternative one can author an explicit model of transfer such as metaphors (Levy and Markovitch 2012) or hypotheses (Kuzborskij and Orabona 2013).

Kuzborskij et al. (2013) investigate the same n to $n+1$ multiclass transfer learning problem as our image classification experiments, and make use of a combination of existing trained classifiers. However, their approach makes use of Support Vector Machines with a small feature-set and only allows for linear combinations. Rebuffi et al. (2017) extended this work to convolutional neural nets, but still requires retraining via backpropagation. Chao et al. (2016) demonstrated that average visual features can be used for zero-shot learning, which represents a domain independent zero-shot learning measure that does not require human authoring or additional training. We employ this last approach as a baseline.

One alternative to reusing learned knowledge in neural networks, is to extend a dataset to new classes using query expansions on the web (Yao et al. 2017). However, we are interested primarily in the question of how existing learned features can be applied to problems in which no additional training data exists, even online, due to the class in question being new, fantastical, or rare. Similarly, Neuroevolution is an approach to train neural networks via evolutionary search, which includes an explicit recombination step (Floreano, Dürr, and Mattiussi 2008). However, this approach does not transfer knowledge from one domain to another.

Conceptual Expansion

Imagine tomorrow we discover that a pegasus exists. Initially we lack enough images of this newly discovered flying horse to build a traditional classifier or image generator. However, suppose we have neural network classifiers and generators trained on classes including horses and birds. Conceptual expansion, a combinational creativity algorithm, allows us to reuse the learned features from machine learned model(s) to produce new models without additional training or additional transfer features.

The intuition behind conceptual expansion is that it defines a high-dimensional, parameterized search space from an arbitrary number of pretrained input models, where each point of this search space is a new model that can be understood as a combination of existing models. We can then explore this space to find models that better meet a goal or heuristic. Each point of this space—each combined model—is a valid conceptual expansion. We can consider

the case where a class or concept (c_X) is a combination of other classes (c_1, \dots, c_n) and that the learned features of models of classes c_1, \dots, c_n can be recombined to create the features of a model of c_X . In these cases, we hypothesize that conceptual expansions can represent models one cannot necessarily discover using conventional machine learning techniques with the available data. Furthermore, we hypothesize that these conceptual expansion models may perform better on specific tasks than standard models in cases with small amounts of available data, such as identifying or generating new classes of objects. In prior work (2018), we demonstrated the application of conceptual expansion to non-neural graphs with hundreds of thousands of edges, which suggests the potential for their application to neural networks. We can use a heuristic informed by this small amount of training data to guide the search for our final conceptual expansion. This process is inspired by the human ability to make conceptual leaps, but is not intended as an accurate recreation.

A conceptual expansion of concept X is represented as the following function:

$$CE^X(F, A) = a_1 * f_1 + a_2 * f_2 \dots a_n * f_n \quad (1)$$

Where $F = \{f_1, \dots, f_n\}$ is the set of all mapped features and $A = \{a_1, \dots, a_n\}$ is a set of filters each dictating what of and what amount of mapped feature f_i should be represented in the final conceptual expansion. In the ideal case $X = CE^X$ (e.g. a combined model of birds and horses equals our ideal pegasus model). The exact shape of a_i depends upon the feature representation. If features are symbolic, a_i can have values of either 0 or 1 (including the mapped feature or not), or vary from 0 to 1 if features are numeric or ordinal. Note that for numeric values one may choose a different range (e.g. -1 to 1) dependent on the domain. If features are matrices, as in a neural net, each a_i is also a matrix. In the case of matrices the multiplication is an element-wise multiplication or Hadamard product. As an example, in the case of neural image recognition, $\{f_1, \dots, f_n\}$ are the variables in a convolutional neural network learned via backpropagation. Deriving a conceptual expansion is the process of finding an A for known features F such that $CE^X(\cdot)$ optimizes a given objective or heuristic towards some target concept X .

In this representation, the space of conceptual expansions is a multidimensional, parameterized search space over possible combinations of our input models. There exists an infinite number of possible conceptual expansions for non-symbolic features, which makes naively deriving this representation ill-advised. Instead, as is typical in combinational creativity approaches, we first derive a *mapping*. The mapping determines what particular prior knowledge—in this case the weights and biases of a neural network—will be combined to address the novel case. This will determine the starting point of the later search process we employ to explore the space of possible conceptual expansions.

Given a mapping, we construct an initial conceptual expansion—a set of $F = \{f_1, \dots, f_n\}$ and an $A = \{a_1, \dots, a_n\}$ —that is iterated upon to optimize for domain specific notions of quality (in the example pegasus case image recognition accuracy). In the following sections we dis-

Algorithm 1: Conceptual Expansion Search

input : available data *data*, an initial model *model*, a mapping *m*, and a score *score*
output: The maximum expansion found according to the heuristic

```

1 maxE ← DefaultExpansion(model) + m;
2 maxScore ← score;
3 v ← [maxE];
4 improving ← 0;
5 while improving < 10 do
6   n ← maxE.GetNeighbor(v);
7   v ← v + n;
8   s ← Heuristic(n, data);
9   oldMax ← maxScore; maxScore, maxE ←
   max([maxScore, maxE], [s, n]);
10  improving ← oldMax < maxScore?0:improving++
11 return maxE;
```

cuss the creation of the mapping and then the refinement of the conceptual expansion.

Mapping Construction

Constructing the initial mapping is relatively straightforward for the purposes of this paper. As input we assume we have an existing trained model or models (CifarNet trained on CIFAR-10 for the purposes of this example (Krizhevsky and Hinton 2009)), and data for a novel class (whatever pegasus images we have). We construct a mapping with the novel class data by examining how the model or models in our knowledge base perform on the data for the novel class. The mapping is constructed according to the ratio of the new images classified into each of the old classes. For example, suppose we have a CifarNet trained on CIFAR-10 and we additionally have four pegasus images. Say CifarNet classifies two of the four pegasus images as a horse and two as a bird. We construct a mapping of: f_1 consisting of the weights and biases associated with the horse class, and f_2 consisting of the weights and biases associated with the bird class. We initialize the A values for both variables to all be 0.5—the classification ratio—meaning a floating point a value for the biases and an a matrix for the weights.

Conceptual Expansion Search

The space of potential conceptual expansions grows exponentially with the number of input features, and the mapping construction stage gives us an initial starting point in this space from which to search. We present the pseudocode for the Conceptual Expansion Search in Algorithm 1. Line 1 creates an initial expansion by combining a default expansion with the mapping information. The exact nature of this depends on the final network architecture. For example, the mapping may overwrite the entirety of the network if the input models and final model have the same architecture or just the final classification layer if not (as in the case of adding an additional class). In this case a default expansion is a conceptual expansion equivalent to the original model(s), in that each variable is replaced by an expanded variable with

its original f_i value and an a_i of 1.0 (or matrix of 1.0's). This means that the initial expansion is functionally identical to the original model, beyond any weights impacted by the mapping. This initial conceptual expansion derived at the end of the mapping construction will be a linear combination of the existing knowledge, but the final conceptual expansion need not be a linear combination.

Once we have a mapping we search for a set of F and A for which the conceptual expansion performs well on a domain-specific measure *Heuristic* (e.g. pegasus classification accuracy). For the purposes of this paper we implement a greedy optimization search that checks a fixed number of neighbors before the search ends. The *GetNeighbor* function randomly selects between one of the following: altering a single element of a single a_i , replacing all of the values of a single a_i replacing values of x_i with a randomly selected alternative x_j , or adding an additional x_i and corresponding random a_i to an expanded variable. The final output of this process is the maximum scoring conceptual expansion found during the search. For the purposes of clarity we refer to these conceptual expansions of neural networks as *combinets*.

CifarNet Experiments

In this section we present a series of experiments meant to demonstrate the strengths and limitations of conceptual expansions for image classification with deep neural networks. We chose CIFAR-10 and CIFAR-100 (Krizhevsky and Hinton 2009) as the domains for this approach as these represent well-understood datasets. It is not our goal to achieve state of the art on CIFAR-10 or CIFAR-100; we instead use these datasets to construct problems in which a system must identify images of a class not present in some initial training set given limited training data on the novel class. We then apply our approach to these problems, comparing them with appropriate baselines. For the source deep neural network model we chose CifarNet (Krizhevsky and Hinton 2009), again due to existing understanding of its performance on the more traditional applications of these datasets. We chose not to make use of a larger dataset like ImageNet or a larger architecture (Deng et al. 2009), as we aim to compare how our approach constructs new features given a limited set of input features. We do not include a full description of CifarNet but note that it is a two-layer convolutional neural net with three fully-connected layers.

For each experiment, we ran our conceptual expansion search algorithm ten times and took the most successful combinet found across the ten runs in terms of training accuracy. We did this to ensure we had found a near optimal conceptual expansion, but anticipate that future work will explore more sophisticated optimization strategies. We note that this approach was still many times faster than initially training the CifarNet on CIFAR-10 with backpropagation.

Our first experiment expands a CifarNet trained on CIFAR-10 to recognize one additional class selected from CIFAR-100 that is not in CIFAR-10. We vary the amount of training data for the newly introduced class. This allows us to evaluate the performance of recombination via

conceptual expansions under a variety of controlled conditions. Our second experiment fully expands a CifarNet model trained on CIFAR-10 to recognize the one-hundred classes of CIFAR-100 with limited training data. Finally, we investigate the running example throughout this paper: expanding a CifarNet model trained on CIFAR-10 to classify pegasus images.

CIFAR-10 + Fox/Plain

For our initial experiment we chose to add fox and plain (as in a grassy field) recognition to the CifarNet, as these classes exist within CIFAR-100, but not within CIFAR-10 (CIFAR-10 is made up of the classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck). We chose foxes and plains for this initial case study because they represented illustrative examples of conceptual expansion performance. There exists a number of classes in CIFAR-10 that can be argued to be similar to foxes, but no classes similar to plains.

For training data we drew from the 50,000 training examples for the ten classes of CIFAR-10, adding a varying number of training instance of fox or plain. For test data we made use of the full 10,000 CIFAR-10 test set and the 100 samples in the CIFAR-100 test set for each class. For each size slice of training data (i.e. 1, 5, 10, 50, and 100) we constructed five unique random slices. We chose five for consistency across all the differently sized slices, given that there was a maximum of 500 training images for fox and plain, and our largest slice size was 100. We present the average test accuracy across all approaches and with all sample sizes in Table 1. This table shows results when we provide five slices of fox or plain images in the quantities of 1, 5, 10, 50, or 100. For each slice, we provide the accuracy on the original CIFAR-10 images and the accuracy of identifying the 11th class (either fox or plains).

We evaluate against three baselines. Our first baseline (standard) trains CifarNet with backpropagation with stratified branches on the 10,000 CIFAR-10 images and newly introduced foxes or plains. This baseline makes the assumption that the new class was part of the same domain as the other classes as in (Daumé III 2009). For our second baseline we took inspiration from transfer learning and student-teacher models (Wong and Gales 2016; Li et al. 2017; Furlanello et al. 2017), and train an initial CifarNet on only the CIFAR-10 data and then retrain the classification layers to predict the eleventh class with the newly available data. We note that transfer learning typically involves training on a larger dataset, such as ImageNet, then retraining the final classification layer. However, we wished to compare how these different approaches alter the same initial features for classifying the new class. For our third baseline we drew on the zero-shot approach outlined in (Chao et al. 2016), using the average activation of the trained CifarNet classification layer on the training data to derive feature classification vectors. In all cases we trained the model until convergence.

There exist many other transfer approaches, but other approaches tend to require additional human authoring of transfer methods or features and/or an additional dataset to draw from. We focus on comparing the behavior of these approaches in terms of altering or leveraging learned features.

Table 1: A table with the average test accuracy for the first experiment. The orig. column displays the accuracy for the 10,000 test images for the original 10 classes of CIFAR-10. The 11th column displays the accuracy for the CIFAR-100 test images.

	100		50		10		5		1	
Fox	11th	orig.	11th	orig.	11th	orig.	11th	orig.	11th	orig.
combinet	34.0±3.5	81.8±2.2	26.0±5.2	81.59±1.9	28.3±3.5	79.1±1.6	23.0±8.5	80.6±1.2	12.0±9.8	80.7±7.2
standard	7.0±2.7	62.04	0.0±0.0	62.17	0.0±0.0	62.34	0.0±0.0	62.44	0.0±0.0	76.44±3.5
transfer	5.0±4.3	87.2±0.5	0.0±0.0	87.9±0.2	0.0±0.0	88.1±0.4	0.0±0.0	87.7±0.2	0.0±0.0	88.0±1.1
zero-shot	11.0±0.7	86.2±0.4	11.0±1.0	86.2±0.8	9.6±2.3	86.2±0.2	10.0±4.6	86.0±1.4	6.0±3.3	83.2±2.5
Plain	11th	orig.	11th	orig.	11th	orig.	11th	orig.	11th	orig.
combinet	53.0±10.0	84.0±3.6	45.7±7.6	84.2±7.8	31.3±22.0	83.9±2.4	28.3±12.6	82.3±2.2	23.0±17.4	84.0±2.4
standard	50.0±7.7	62.54	42.0±3.2	62.18	16.0±12.8	61.67	0.0±0.0	62.27	0.0±0.0	62.27
transfer	4.5±3.0	86.92	0.0±0.0	86.91	0.0±0.0	86.96	0.0±0.0	87.20	0.0±0.0	87.20
zero-shot	23.0±0.7	86.2±0.5	23.6±1.1	86.2±0.3	22±2.8	86.1±13.9	18.6±3.8	83.7±3.4	15.6±7.3	82.7±2.9

As can be seen in Table 1, the combinet consistently outperforms the baselines at recognizing the newly added eleventh class. We note that the expected CifarNet test accuracy for CIFAR-10 is 85%. Combinets achieve the best accuracy on the newly added class while only losing a small amount of accuracy on average on the 10 original classes. The combinet loss in CIFAR-10 accuracy was almost always due to overgeneralizing. The transfer approach did slightly better than the expected CIFAR-10 accuracy, but this matches previously reported accuracy improvements from retraining (Furlanello et al. 2017).

Foxes clearly confused the baselines, leading to no correctly identified test foxes for the standard or transfer baselines at the lowest values. Compared to plains, foxes had significant overlap in terms of features with cats and dogs. With these smaller size samples transfer and standard were unable to learn or adapt suitable discriminatory features. Comparatively, the conceptual expansion approach was capable of combining existing features into new features that were more successfully able to discriminate between these classes. The zero-shot approach did not require additional training and instead made use of secondary features to make predictions, which was more consistent, but still not as successful as our approach in classifying the new class. In comparison plain was much easier to recognize for our baselines, likely due to the fact that it represented a class that differed significantly from the existing ten. However, our approach was still able to outperform this, creating novel features that could better differentiate the plain class.

Note that combinet does not always outperform these other approaches. For example, the standard approach beats out combinet, getting an average of 83% accuracy with access to all 500 plain training images, while the combinet only achieves an accuracy of roughly 50%. This suggests that combinet is only suited to problems with low training data with this current approach.

Expanding CIFAR-10 to CIFAR-100

For the prior experiments we added a single eleventh class from CIFAR-100 to a CifarNet trained on CIFAR-10. This experiment looks at the problem of expanding a trained CifarNet from classifying the ten classes of the CIFAR-10 dataset to the one-hundred classes of the CIFAR-100 dataset.

For this experiment we limited our training data to ten ran-

domly chosen samples of each CIFAR-100 class. We altered our approach to account for the change in task, constructing an initial mapping for each class individually as if we were expanding a CifarNet to just that eleventh class. We utilized the same three baselines as with the first experiment.

We note that one would not typically utilize CifarNet for this task. Even given access to all 50,000 training samples of CIFAR-100 a CifarNet trained using backpropagation only achieves roughly 30% test accuracy for CIFAR-100. We mean to show the relative scale of accuracy before and after conceptual expansion and not an attempt to achieve state of the art on CIFAR-100 with the full dataset. We tested on the 100,000 test samples available for CIFAR-100.

The average test accuracy across all 100 classes are as follows: the combinet achieves 11.13%, the standard baseline achieves 1.20%, the transfer baseline achieves 6.43%, and the zero-shot baseline achieves 4.10%. We note that our approach is the only one to do better than chance, and significantly outperforms all other baselines. However no approach reaches anywhere near the 30% accuracy that could be achieved with full training data for this architecture.

Pegasus

We return to our running example of an image recognition system that can recognize a pegasus. Unfortunately we lack actual images of a pegasus. To approximate this we collected fifteen photo-realistic, open-use pegasus images from Flickr. Using the same combinet as the above two experiments we ran a 10-5 training/test split and a 5-10 training/test split. For the former we recognized 4 of the 5 pegasus images (80% accuracy), with 80% CIFAR-10 accuracy, and for the latter we recognized 5 of the 10 pegasus images (50% accuracy) with 82% CIFAR-10 accuracy.

DCGAN Experiment

In this section we demonstrate the application of conceptual expansions to generative adversarial networks (GANs). Specifically, we demonstrate the ability to use conceptual expansions to find GANs that can generate images of a class without traditional training on images of that class. We also demonstrate how our approach can take as input an arbitrary number of initial neural networks, instead of the one network for the classification experiments. We make use

Table 2: Summary of results for the GAN experiments.

Samples	combiGAN		combi+N		combi+T		Naive		Transfer	
	I	KL	I	KL	I	KL	I	KL	I	KL
500	3.83±0.32	0.33	4.61±0.22	0.28	3.05±0.23	0.31	2.98±0.25	0.33	3.38±0.19	1.05
100	4.23±0.15	0.10	4.38±0.37	0.29	4.40±0.19	0.43	1.76±0.04	0.33	3.26±0.23	0.36
50	4.05±0.24	0.22	4.03±0.35	0.12	1.69±0.05	2.36	1.06±0.00	10.8	3.97±0.22	0.21
10	4.67±0.44	0.44	4.79±0.28	0.13	3.06±0.19	1.20	1.20±0.01	10.8	4.40±0.19	0.11

of the DCGAN (Radford, Metz, and Chintala 2015) as the GAN architecture for this experiment, as it has known performance on a number of tasks. We make use of the CIFAR-100 dataset from the prior section and in addition use the Caltech-UCSD Birds-200-2011 (Wah et al. 2011), the CAT (Zhang, Sun, and Tang 2008), the Stanford Dogs (Khosla et al. 2011), FGVC Aircraft (Maji et al. 2013), and the Stanford Cars (Krause et al. 2013) datasets. We make use of these five datasets as they represent five of the ten CIFAR-10 classes, but with significantly more images and images of higher quality. Sharing the majority of classes between experiments allows us to draw comparisons between results.

We trained a DCGAN on each of these datasets till convergence, then used all five of these models as the original knowledge base for our approach. Specifically, we built mappings by testing the proportion of training samples the discriminator of each GAN classified as real. We then built a combinet discriminator for the target class from the discriminators of each GAN. Finally we built a combinet generator from the generators of each GAN, using the combinet discriminators as the heuristic for the conceptual expansion search. We nickname these combinet discriminators and generators *combiGANs*. As above we made use of the fox images of CIFAR-100 for the novel class, varying the number of available images.

We built two baselines: (1) A naive baseline, which involved training the DCGAN on the available fox images in the traditional manner. (2) A transfer baseline, in which we took a DCGAN trained on the Stanford Dogs dataset and retrained it on the fox dataset. We also built two variations of combiGAN: (1) A combiGAN baseline in which we used the discriminator of the naive baseline as the heuristic for the combinet generator (Combi+N). (2) Same as the last, but using the transfer baseline discriminator (Combi+T). We further built a baseline trained on the Stanford Dogs, CAT dataset, and Fox images simultaneously as in (Cheong and Teo 2018), but found that it did not have any improvement over the other baselines. We omit it to save space. We do not include the zero shot baseline from the prior section as it is only suitable for classification tasks.

CombiGAN Results

We made use of two metrics: the inception score (Salimans et al. 2016) and Kullback-Leibler (KL) divergence between generated image classification and true image classification distributions. We acknowledge that inception score was originally designed for ImageNet; since we do not train on ImageNet, we cannot use this as an objective score, but we

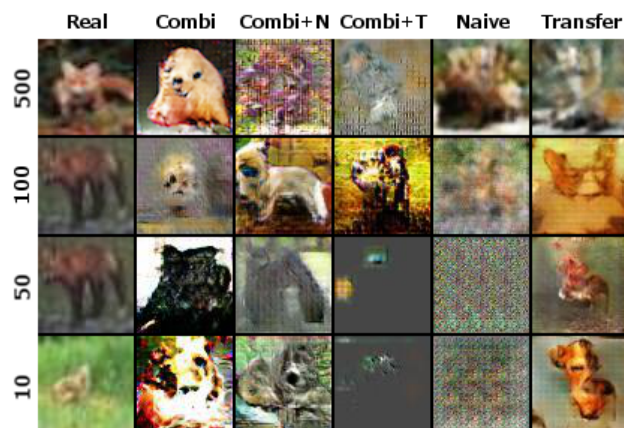


Figure 1: Most fox-like output according to our model for each baseline and sample size.

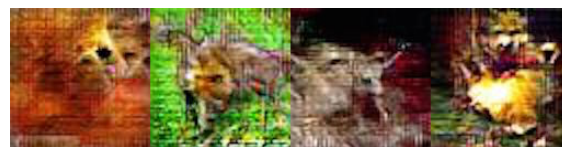


Figure 2: Four fox-like images hand-picked by the authors from the first 1,000 images output by the combiGAN trained on 500 foxes.

can use it as a comparative metric of objectness. For the second metric we desired some way to represent how fox-like the generated images were. Thus we made use of the standard classifier trained on 500 foxes, though we could have made use of any classifier in theory. We compare the distribution over classes of real CIFAR-100 fox images and the fake images with the KL divergence. We generated 10,000 images from each GAN to test each metric. We summarize the results of this experiment in Table 2.

We note that in almost all cases our approach or one of its variations (combi+N and combi+T) outperform the two baselines. In the case with 10 training images the transfer baseline beats our approach on our fox-like measure, but this 0.11 differs only slightly from the 0.13 combi+N value. In Figure 1, we include the most fox-like image in terms of classifier confidence from the training samples (real) and each baseline's output. We note that the combiGAN output had a tendency to retain face-like features, while the transfer baseline tended to revert to fuzzy blobs. We also include

four hand-picked fox-like images from the 500 sample case in Figure 2.

Discussion and Limitations

Conceptual expansions of neural networks—combinets and combiGANs—outperform standard approaches on problems with limited data without additional knowledge engineering. We refer to this approach generally as conceptual expansion, which is inspired by the human ability to make conceptual leaps by combining existing knowledge. Our main contribution in this paper is the initial exploration of conceptual expansion of neural networks; we speculate that more sophisticated optimization search routines may achieve greater improvements.

We anticipate the future performance of conceptual expansions to depend upon the extent to which the existing knowledge base contains relevant information to the new problem and ability for the optimization function to find helpful conceptual expansions. We note that one choice of optimization function could be human intuition, and we have had success hand-designing conceptual expansions for sufficiently small problems.

Conceptual expansions appear less dependent on training data than existing transfer learning approaches as evidenced by the comparative performance of the approach with low training data. This is further evidenced by those instances where conceptual expansion outperformed itself with less training data. We anticipate further exploration of this in future work.

Conclusions

We present conceptual expansion of neural networks: *combinets*, an approach to produce recombined versions of existing machine learned deep neural net models. We ran four experiments of this approach compared to common baselines, and found we were able to achieve greater accuracy with less data. Our technique relies upon a flexible representation of recombination of existing knowledge that allows us to represent new knowledge as a combination of particular knowledge from existing cases. To our knowledge this represents the first attempt at applying a model of combinational creativity to neural networks.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. IIS-1525967. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Ba, L. J.; Swersky, K.; Fidler, S.; and Salakhutdinov, R. 2015. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *International Conference on Computer Vision*, 4247–4255.

Boden, M. A. 2004. *The creative mind: Myths and mechanisms*. Psychology Press.

Boden, M. A. 2009. Computer models of creativity. *AI Magazine* 30(3):23–23.

Chao, W.-L.; Changpinyo, S.; Gong, B.; and Sha, F. 2016. An empirical study and analysis of generalized zero-shot learning for object recognition in the wild. In *European Conference on Computer Vision*, 52–68. Springer.

Cheong, B., and Teo, H. 2018. Can we train dogs and humans at the same time? gans and hidden distributions. Technical report, Stanford.

Cojan, J., and Lieber, J. 2009. Belief merging-based case combination. In *ICCB*, 105–119. Springer.

Cunha, J. M.; Gonçalves, J.; Martins, P.; Machado, P.; and Cardoso, A. 2017. A pig, an angel and a cactus walk into a blender: A descriptive approach to visual blending. *arXiv preprint arXiv:1706.09076*.

Daumé III, H. 2009. Frustratingly easy domain adaptation. *arXiv preprint arXiv:0907.1815*.

De Mantaras, R. L.; McSherry, D.; Bridge, D.; Leake, D.; Smyth, B.; Craw, S.; Faltings, B.; Maher, M. L.; T COX, M.; Forbus, K.; et al. 2005. Retrieval, reuse, revision and retention in case-based reasoning. *The Knowledge Engineering Review* 20(3):215–240.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR09*.

Elhoseiny, M.; Zhu, Y.; Zhang, H.; and Elgammal, A. 2017. Zero shot learning from noisy text description at part precision. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Fauconnier, G. 2001. Conceptual blending and analogy. *The analogical mind: Perspectives from cognitive science* 255–286.

Fei-Fei, L.; Fergus, R.; and Perona, P. 2006. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence* 28(4):594–611.

Floreano, D.; Dürr, P.; and Mattiussi, C. 2008. Neuroevolution: from architectures to learning. *Evolutionary Intelligence* 1(1):47–62.

Fox, J., and Clarke, S. 2009. Exploring approaches to dynamic adaptation. In *Proceedings of the 3rd International DiscCoTec Workshop on Middleware-Application Interaction*, 19–24. ACM.

Furlanello, T.; Lipton, Z. C.; Amazon, A.; Itti, L.; and Anandkumar, A. 2017. Born again neural networks. In *NIPS Workshop on Meta Learning*.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research* 17(1):2096–2030.

Guzdial, M., and Riedl, M. 2016. Learning to blend computer game levels. In *Seventh International Conference on Computational Creativity*.

Guzdial, M., and Riedl, M. 2018. Automated game design

- via conceptual expansion. In *Fourteenth Artificial Intelligence and Interactive Digital Entertainment Conference*.
- Hervás, R., and Gervás, P. 2006. Case-based reasoning for knowledge-intensive template selection during text generation. In *European Conference on Case-Based Reasoning*, 151–165. Springer.
- Khosla, A.; Jayadevaprakash, N.; Yao, B.; and Fei-Fei, L. 2011. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*.
- Konieczny, S., and Pérez, R. P. 2011. Logic based merging. *Journal of Philosophical Logic* 40(2):239–270.
- Krause, J.; Stark, M.; Deng, J.; and Fei-Fei, L. 2013. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*.
- Krizhevsky, A., and Hinton, G. 2009. Learning multiple layers of features from tiny images. *Citeseer*.
- Kulis, B.; Saenko, K.; and Darrell, T. 2011. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 1785–1792. IEEE.
- Kuzborskij, I., and Orabona, F. 2013. Stability and hypothesis transfer learning. In *International Conference on Machine Learning*, 942–950.
- Kuzborskij, I.; Orabona, F.; and Caputo, B. 2013. From n to n+ 1: Multiclass transfer incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3358–3365.
- Lampert, C. H.; Nickisch, H.; and Harmeling, S. 2009. Learning to detect unseen object classes by between-class attribute transfer. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 951–958. IEEE.
- Levy, O., and Markovitch, S. 2012. *Teaching machines to learn by metaphors*. Technion-Israel Institute of Technology, Faculty of Computer Science.
- Li, B.; Zook, A.; Davis, N.; and Riedl, M. O. 2012. Goal-driven conceptual blending: A computational approach for creativity. In *Proceedings of the 2012 International Conference on Computational Creativity, Dublin, Ireland*, 3–16.
- Li, J.; Seltzer, M. L.; Wang, X.; Zhao, R.; and Gong, Y. 2017. Large-scale domain adaptation via teacher-student learning. *arXiv preprint arXiv:1708.05466*.
- Maji, S.; Kannala, J.; Rahtu, E.; Blaschko, M.; and Vedaldi, A. 2013. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*.
- Manzano, S.; Ontañón, S.; and Plaza, E. 2011. Amalgam-based reuse for multiagent case-based reasoning. In *International Conference on Case-Based Reasoning*, 122–136. Springer.
- Mensink, T.; Gavves, E.; and Snoek, C. G. 2014. Costa: Co-occurrence statistics for zero-shot classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2441–2448.
- Norouzi, M.; Mikolov, T.; Bengio, S.; Singer, Y.; Shlens, J.; Frome, A.; Corrado, G. S.; and Dean, J. 2013. Zero-shot learning by convex combination of semantic embeddings. *arXiv preprint arXiv:1312.5650*.
- Ontañón, S., and Plaza, E. 2010. Amalgams: A formal approach for combining multiple case solutions. In *Case-Based Reasoning. Research and Development*. Springer. 257–271.
- Pereira, F.; Norvig, P.; and Halevy, A. 2009. The unreasonable effectiveness of data. *IEEE Intelligent Systems* 24(undefiend):8–12.
- Radford, A.; Metz, L.; and Chintala, S. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.
- Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; and Lampert, C. H. 2017. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.
- Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; and Chen, X. 2016. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, 2234–2242.
- Sizov, G.; Öztürk, P.; and Aamodt, A. 2015. Evidence-driven retrieval in textual cbr: bridging the gap between retrieval and reuse. In *International Conference on Case-Based Reasoning*, 351–365. Springer.
- Thagard, P., and Stewart, T. C. 2011. The aha! experience: Creativity through emergent binding in neural networks. *Cognitive science* 35(1):1–33.
- Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology.
- Wang, Y.-X., and Hebert, M. 2016. Learning to learn: Model regression networks for easy small sample learning. In *European Conference on Computer Vision*, 616–634. Springer.
- Wilke, W., and Bergmann, R. 1998. Techniques and knowledge used for adaptation during case-based problem solving. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, 497–506. Springer.
- Wong, J. H., and Gales, M. J. 2016. Sequence student-teacher training of deep neural networks. *International Speech Communication Association*.
- Xian, Y.; Schiele, B.; and Akata, Z. 2017. Zero-shot learning-the good, the bad and the ugly. *arXiv preprint arXiv:1703.04394*.
- Yao, Y.; Zhang, J.; Shen, F.; Hua, X.; Xu, J.; and Tang, Z. 2017. Exploiting web images for dataset construction: A domain robust approach. *IEEE Transactions on Multimedia* 19(8):1771–1784.
- Zhang, W.; Sun, J.; and Tang, X. 2008. Cat head detection-how to effectively exploit shape and texture features. In *European Conference on Computer Vision*, 802–816. Springer.

Trick or : Thematic Reinforcement for Artistic Typography

Purva Tendulkar¹ Kalpesh Krishna² Ramprasaath R. Selvaraju¹ Devi Parikh¹

¹Georgia Institute of Technology, ²University of Massachusetts, Amherst
¹{purva, ramprs, parikh}@gatech.edu, ²kalpesh@cs.umass.edu

Abstract

An approach to make text visually appealing and memorable is *semantic reinforcement* – the use of visual cues alluding to the context or theme in which the word is being used to reinforce the message (e.g., Google Doodles). We present a computational approach for semantic reinforcement called TReAT – Thematic Reinforcement for Artistic Typography. Given an input word (e.g. *exam*) and a theme (e.g. *education*), the individual letters of the input word are replaced by cliparts relevant to the theme which visually resemble the letters – adding creative context to the otherwise ordinary and uninspiring input word. We use an unsupervised approach to learn a latent space to represent letters and cliparts and compute similarities between the two. Human studies show that participants can reliably recognize the word as well as the theme in our outputs (TReATs) and find them more creative compared to meaningful baselines.

Introduction

We address the task of theme-based word typography: given a word (e.g., *exam*) and a theme (e.g., *education*), the task is to *automagically* produce a “doodle”¹ for the word in that theme as seen in Figure 1. Concretely, the task is to replace each letter in the input word with a clipart from the input theme to produce a doodle, such that the word and theme can be easily identified from the doodle. Solving this task would be of value to a variety of creative applications such as stylizing text in advertising, designing logos – essentially any application where a message needs to be conveyed to an audience in an effective and concise manner.

Using graphic elements to emphasize the meaning of a word in reference to a related theme is referred to in graphic design as *semantic reinforcement*. This can be achieved in a number of ways, e.g., using different fonts and colors (Figure 2a), changing the position of letters relative to one another (Figure 2b), arranging letters in a specific direction or shape (Figure 2c), excluding some letters (Figure 2d), adding icons near or around the letters (Figure 2e), or replacing letters with icons (Figure 2f). In our work, we focus on this last type, i.e., semantic reinforcement via replacement.

¹In this work, we use “doodle” to refer to Google doodles-like typography as in Figure 2f.

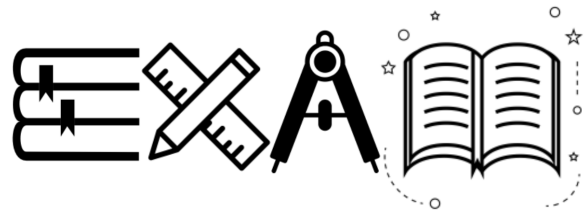


Figure 1: A sample doodle (that we call TReAT) generated by our system for the input word *exam* and theme *education*².

This is a challenging task even for humans. It not only requires domain-specific knowledge for identifying a set of relevant cliparts to choose from, but also requires creative abilities to be able to visualize a letter in a clipart, and choose the best clipart for representing it.

The latter alone is challenging to automate – both from a training and evaluation perspective. Training a model to automatically match letters to graphics is challenging because there is a lack of large-scale text-graphic paired datasets in each domain that might be of interest (e.g., clipart, logogram). Evaluation and thus iterative development of such models is also challenging because of subjectivity and inter-human disagreement on which letter resembles which graphic.

We present a computational approach – Thematic Reinforcement of Artistic Typography (TReAT) – to generate doodles (TReATs) for semantic reinforcement of text via replacement. We represent letters in different fonts and cliparts from the Noun Project. in a common latent space. These latent representations are learned such that they have two characteristics: (1) The letters can be correctly recognized (e.g., *a* vs. *b*) in the latent space and (2) The letters and cliparts can be reconstructed accurately from the latent space. A reconstruction loss ensures that letters and clipart that are close in the latent space also look similar in the image space. A classification loss ensures that the latent space is informed by discriminative features that make one letter different from the other. This allows us to match cliparts to letters in a way that preserves distinctive visual features of the letters, making it easier for humans to identify the letter being depicted by a

²Unless stated otherwise, all cliparts in the paper have been taken from The Noun Project - <https://thenounproject.com/>. The Noun Project contains cliparts created by different graphic designers on a variety of themes.



Figure 2: Different methods for semantic reinforcement. a) font and color variations b) positioning of letters relative to each other c) arrangement of letters in a specific shape or direction d) exclusion of some letters e) addition of icons near letters f) replacement of letters. In this work we focus on f), semantic reinforcement via replacement.³

clipart. At test time, given a word and a theme as input, we first retrieve cliparts from the Noun Project that match that theme. For each letter in the word, we find the theme-relevant clipart which minimizes the distance from it across a variety of fonts. If the distance is low enough, we replace the letter with the clipart.

We run human studies to show that subjects can reliably recognize the word as well as the theme from our TReATs, and find them to be creative. We consider a TReAT to be creative if it is surprising and/or intriguing and/or fun.

Related work

Early human communication was through symbols and hieroglyphs (Frutiger 1989), (Schmandt-Besserat 2015). This involved the use of characters to represent an entire word, phrase or concept. Then language evolved and we started using the alphabet for creation of new words to represent concepts. However many languages (e.g. Chinese and Japanese) still make use of pictograms or logograms to depict specific words. Today, symbols and logos are used for creative applications to increase the communication bandwidth – to convey abstract concepts, express rich emotions, or reinforce messages (Shiojiri and Nakatani 2013), (Clawson et al. 2012), (Takasaki and Mori 2007). Our work produces a visual depiction of text by reasoning about similarity between the visual appearance of a letter and clipart imagery. We describe prior work in each of these domains: creativity through imagery and creativity through visual appearance of text.

Creativity through imagery

There has been work on evoking emotional responses through the modification of images. Work on visual blending of emojis combines different concepts to create novel emojis (Cunha et al. 2018). Visual blending has also been explored for combining two animals to create images depicting fictional hybrid animals (Martins et al. 2015). Our approach tries to induce creativity by entirely replacing a letter with a clipart.

Work on Vismantic (Xiao and Linkola 2015) represents abstract concepts visually by combining images using juxtaposition, fusion, and replacement. Our work also represents a theme via replacement (replacing letters with cliparts); however our replacement is for the purposes of lexical resolution, not visual. Recently, there has been an exploration of neural style transfer for logo generation (Atarsaikhan, Iwana, and Uchida 2018). This work however only transfers color and texture from the style to the content. In our work, the

³Examples a) to e) were taken from this answer on StackExchange. Example f) is a Google Doodle.

transfer occurs via direct replacement. Logo generation has also been explored through the use of Generative Adversarial Networks (Sage et al. 2018).

Recently Google’s QuickDraw! and AutoDraw based on sketch-rnn (Ha and Eck 2018) have gained a lot popularity. Their work trains a recurrent neural network (RNN) to construct stroke-based drawings of common objects, and is also able to recognize objects from human strokes. One could envision creating a doodle by writing out one letter at a time, that AutoDraw would match to the closest object in its library. However, these matches would not be theme based. Iconary⁴ is a very recent pictiary-like game that users can play with an AI. Relevant to this work, user drawings in Iconary are mapped to icons from The Noun Project to create a scene.

The use of conditional adversarial networks for image-to-image translation is gaining popularity. However, using a pix2pix-like architecture (Isola et al. 2017) for our task would involve the use of labeled pairwise (clipart, letter) data, which as discussed earlier, is hard to obtain. CycleGAN (Zhu et al. 2017) does not require paired label data, but is not a good fit for our task because we are interested in matching letters to cliparts from a specific theme. The pool of clipart is thus limited, and would not be sufficient to learn the target domain. Finally, generative modeling is typically lossy; we prefer direct replacement of cliparts for greater readability.

Creativity through visual appearance of text

Advances in conditional Generative Adversarial Networks (cGANs) (Mirza and Osindero 2014) have motivated style transfer for fonts (Azadi et al. 2018) through few-shot learning. Work on learning a manifold of fonts (Campbell and Kautz 2014) allows everyday users to create and edit fonts by smoothly interpolating between existing fonts. (Martins et al. 2018) explore an evolutionary system for the generation of type stencils constructed from line segments. The first work explores the creation of unseen letters of a known font, while the other two works explore the creation of entirely new fonts – neither add any theme-related semantics or additional graphic elements to the text.

Work on neural font style transfer between fonts (Atarsaikhan et al. 2017) explores the effects of using different weighted factors, character placements and orientations in style transfer. This work also has an experiment using icons as style images, however the style transfer is only within the context of visual features of icons such as the texture and thickness of strokes, as opposed to direct replacement.

⁴<https://iconary.allenai.org/>

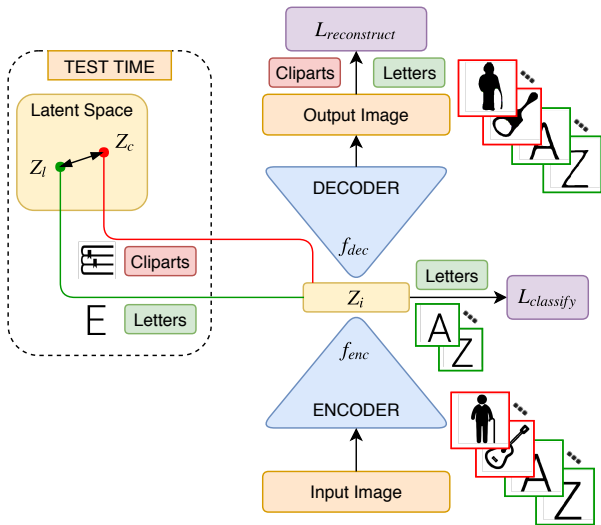


Figure 3: Block diagram describing our model during training and testing. The model is trained on a reconstruction and classification loss in a multitask fashion. During inference, latent space distances are calculated to match letters to cliparts. See text for more details.

MarkMaker⁵ generates logos based on company names – primarily displaying the name in various fonts and styles, sometimes along with a clipart. It uses a genetic algorithm to iteratively refine suggestions based on user feedback.

Approach

In this section, we first describe our procedure for collecting training data, and then our model and its training details. Finally, we describe our test-time procedure to generate a TReAT, that is, obtaining theme-based clipart matches for an input word and theme. A sketch of our model along with examples for training and testing are shown in Figure 3.

Training Data

For our task we need two types of data for training – letters in different fonts, and cliparts. Note that we do not need a *correspondence* between the letters and cliparts. In that sense, as stated earlier, our approach is an unsupervised one.

For clipart data, we use the Noun Project – a website that aggregates and categorizes symbols that are created and uploaded by graphic designers around the world. The Noun Project cliparts are all 200×200 in PNG format. The Noun Project has binary cliparts, and will result in TReATs of the style shown in Figure 1. Different choices of the source of cliparts can result in different styles, including colored TReATs similar to Figure 2f. We downloaded a random set of $\sim 50k$ cliparts from the Noun Project.

We obtain our letter data from a collection of 1400 distinct font files.⁶ On manual inspection, we found that this set contained a lot of visual redundancies (e.g. the same font being repeated in regular and bold weight types). We removed such repetitions. We also manually inspected the data to ensure

⁵<https://emblematic.org/markmaker/>

⁶These font files (TTF) were obtained from a designer colleague.

that the individual letters were recognizable in isolation, and discarded overly complicated and intricate font styles. This left us with a total of 777 distinct fonts. We generated 200×200 image files (PNG format) from each font file for the entire alphabet (uppercase and lowercase) giving us a total of 40.4k images of letters ($777 \text{ fonts} \times 26 \text{ letters in the English alphabet} \times 2 \text{ (upper and lower cases)}$).

Model

Our primary objective is to find visual similarities between cliparts and letters in an unsupervised manner. To this end, we train an autoencoder (Ballard 1987) with a reconstruction loss on both clipart and letter images (denoted by \mathbf{X}_{cl}). We denote a single input image by X_i . Each input image X_i is passed through an encoder neural network $f_{enc}(\cdot)$ and projected to a low dimensional intermediate representation Z_i . Finally, a decoder neural network $f_{dec}(\cdot)$ tries to reconstruct the input image as \hat{X}_i , using the objective $\mathcal{L}_{reconstruct}$,

$$Z_i = f_{enc}(X_i) \quad (1)$$

$$\hat{X}_i = f_{dec}(Z_i) \quad (2)$$

$$\mathcal{L}_{reconstruct} = \frac{1}{|\mathbf{X}_{cl}|} \sum_{i \in \mathbf{X}_{cl}} \text{SSD}(X_i, \hat{X}_i) \quad (3)$$

where $\text{SSD}(X_i, \hat{X}_i)$ is the sum over squared pixel differences between the original image and its reconstruction. We set the dimensionality of Z_i to be 128. In addition to the reconstruction objective, we utilize letter labels (52 labels for lowercase and uppercase letters) to classify the intermediate representations Z_i for the letter images. This objective helps the encoder discriminate between different letters (possibly with similar visual features) while clustering together the intermediate representations for the same letter across different fonts. This would allow the intermediate representation to capture visual features that are characteristic of each letter, and when cliparts are matched to letters using this representation, the matched cliparts will retain the visually discriminative features of letters.

Concretely, we project Z_i to a 52-dimensional space using a single linear layer with a softmax non-linearity and use the cross entropy loss function. Let W and b be the parameters of a linear transformation of Z_i . We obtain a probability distribution $P_i(\cdot)$ across all labels as,

$$P_i(\cdot) = \text{softmax}(WZ_i + b) \quad (4)$$

Let \mathbf{X}_l be the subset of images in \mathbf{X}_{cl} that are letters. We maximize the probability of the correct label Y_i corresponding to each letter image X_i .

$$\mathcal{L}_{classify} = -\frac{1}{|\mathbf{X}_l|} \sum_{i \in \mathbf{X}_l} \log P_i(Y_i) \quad (5)$$

Note that the same Z_i is used in both objective functions for letter images. These objectives are jointly trained using a multitask objective

$$\mathcal{L} = \alpha \mathcal{L}_{reconstruct} + (1 - \alpha) \mathcal{L}_{classify} \quad (6)$$

Our final loss function is thus composed of two different loss functions: (1) $\mathcal{L}_{reconstruct}$ trained on both letters and



Figure 4: Example TReATs generated by our approach for (word & theme) pairs: a) (canoe & watersports) b) (world & countries, continents, natural wonders) c) (water & drinks) d) (church & priest, nun, bishop)

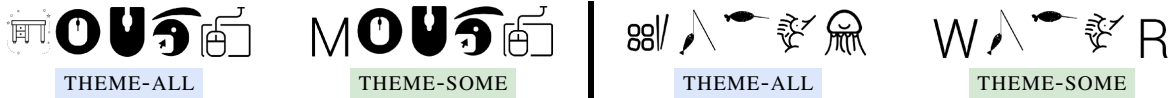


Figure 5: We replace letters in a word with cliparts only if the clipart is sufficiently similar to the letter, placing more stringent conditions on the first and last letters in the word. Notice that in each pair, the TReATs on the right (with a subset of letters replaced) are more legible (Mouse and Water) than the ones on the left, while still depicting the associated themes (computer and fish, mermaid, sailor).

cliparts, and (2) $\mathcal{L}_{\text{classify}}$ trained only on letters. Here α is a tunable hyperparameter in the range $[0, 1]$. We set α to 0.25 after manually inspecting outputs of a few word-theme pairs we used while developing our model (different from the word-theme pairs we use to evaluate our approach later).

Implementation Details: Our encoder network $f_{\text{enc}}(\cdot)$ is an AlexNet (Krizhevsky, Sutskever, and Hinton 2012) convolutional neural network trained from scratch, made up of 5 convolutional and 3 fully connected layers. Our decoder network f_{dec} consists of 5 deconvolutional layers, 3 fully connected layers and 3 upsampling layers. We use batch norm between layers.⁷ We use ReLU activations for both the encoder and decoder. We use the Adam optimizer with a learning rate of 10^{-4} , and a weight decay of 10^{-5} . The input dataset is divided into minibatches of size 100 with a mixture of clipart and letter images in each minibatch. We use early stopping based on a validation set as our stopping criterion.

Data Preprocessing: We resize our images to 224×224 using bilinear interpolation to match the input size of our AlexNet-based encoder. We normalize every channel of our input data to fall in $[-1, 1]$.

Finding Matches

At test time given a word and a theme, we retrieve a theme-relevant pool of cliparts (denoted by \mathbf{C}) by querying Noun Project. These theme-relevant cliparts may or may not be part of the randomly downloaded $\sim 50k$ cliparts used during training. If multiple phrases have been used to describe a theme, we use each phrase separately as a query. We limit this retrieval to no more than 10,000 cliparts for each phrase. We then combine cliparts for different phrases of a theme together to form the final pool of cliparts for that theme. For example, for the theme *countries, continents, natural wonders*, we query Noun Project for *countries, continents and natural wonders* individually and combine all retrieved cliparts together to form the final pool of theme-relevant cliparts. On average across 95 themes we experimented with, we had a minimum of 49 and maximum of 29,580 cliparts per theme, with a mean of 9731.2 and median of 9966. We augmented this set of cliparts with left-right (mirror) flips of the cliparts.

⁷Implementations of our encoder and decoder were adapted from <https://github.com/arnaghosh/Auto-Encoder>.

This improves the overall match quality. E.g., in cases where there existed a good match for \mathbb{J} , but not for \mathbb{L} , the clipart match for \mathbb{J} , when flipped, served as a good match for \mathbb{L} . Similarly for \mathbb{S} and \mathbb{Z} .

For each letter l of the input word, we choose the corresponding letter images (denoted by \mathbf{F}_l) taken from a predefined pool of fonts. To create the pool of letter images \mathbf{F}_l , we used uppercase letters from 14 distinct, readable fonts from among the 777 fonts used during training. These were kept fixed for all experiments. We found that uppercase letters had better matches with the cliparts (lower cosine distance between corresponding latent representations Z_i on average, and visually better matches). Moreover, we found that in several fonts, letters were the same for both cases.

We replace the letter with a clipart (chosen from \mathbf{C}) whose mean cosine distance in the intermediate latent space is the least, when computed against every letter image in \mathbf{F}_l . Concretely, if Z_i denotes the intermediate representation for the i^{th} image in \mathbf{F}_l and Z_c is the intermediate representation for a clipart c in \mathbf{C} , the chosen clipart \hat{c}_l is

$$\hat{c}_l = \operatorname{argmin}_{c \in \mathbf{C}} \frac{1}{|\mathbf{F}_l|} \sum_{i \in \mathbf{F}_l} \left(1 - \frac{Z_i \cdot Z_c}{|Z_i||Z_c|} \right) \quad (7)$$

We find a clipart that is most similar to the letter on average across fonts to ensure a more robust match than considering a single most similar font. In this way, each letter in the input word is replaced by its closest clipart to generate a TReAT.

We show example TReATs generated by our approach in Figure 4. We find that the word can often be difficult to recognize from the TReAT if the Noun Project cliparts corresponding to a theme are not sufficiently similar to the letters in the word. To improve the legibility of our TReATs, we first normalize the cosine distance values of our matched cliparts for the alphabet for a specific theme in the range $[0, 1]$. We only replace a letter with its clipart match if the normalized cosine distance between the embedding of the letter and clipart is < 0.75 . It is known that the first and last letters of a word play a crucial role in whether humans can recognize the word at a glance. (Rawlinson 2007) So we use a stricter threshold, and replace the first and last letters of a word with a clipart only if the normalized cosine distance between the two is < 0.45 . Example TReATs with all letters replaced and only a subset of letters replaced can be seen in Figure 5. Clearly, the TReATs with a subset of letters

replaced (**THEME-SOME**) are more legible than replacing all letters (**THEME-ALL**), while still depicting the desired theme. We quantitatively evaluate this in the next section.

Evaluation

We evaluate our entire system along three dimensions:

- How well is our model able to learn a representation that captures visual features of the letters?
- How does our chosen source of cliparts (Noun Project) affect the quality of matches?
- How good are our generated TReATs?

Learnt Representation

We use t-Distributed Stochastic Neighbor Embedding (t-SNE) to visualize our learnt latent representations of letters and cliparts. Among letters, we find that our model clusters letters of different fonts together, while distinguishing between visually dissimilar letters. E.g., Figure 7 visualizes in 2D uppercase O, Q, E and F in the 14 fonts used at test time. As expected, O and Q clusters are close, and E and F clusters are close, but both these sets of clusters are apart. Visualizing letters as well as cliparts, Figure 8 shows that our model is able to learn a representation such that visually similar letter-clipart pairs are close in the latent space.

Effect of source of cliparts

Themes which have fewer cliparts, and hence typically lower diversity and coverage across the letters (e.g. *mythical beast* in Figure 9a) have poorer matches as compared to larger, more diverse themes (e.g. *library* in Figure 9b). Indeed, we see that recognizing the word in a TReAT generated from the former theme is significantly harder than for the latter. Note that the diversity of cliparts is more important than the quantity. Fewer but more diverse cliparts can lead to better TReATs than many but less diverse cliparts.

Quality of TReATs

We now evaluate the quality of TReATs generated by our approach. We developed our approach on a few themes (e.g., *education*, *Harry Potter*, *Halloween*, *Olympics*) and associated words (e.g., *exam*, *always*, *witch*, *play*). We select these arbitrarily as diverse and popular domains. To evaluate our approach in the open world, we collected 104 word-theme pairs from subjects on Amazon Mechanical Turk (AMT). We told subjects that given a word and an associated theme, we have a bot that can draw a doodle. We showed subjects a few example TReATs. We asked subjects to give us a word and an associated theme (to be described in 1-5 comma separated phrases) that they would like to see a doodle for. Example (word & theme) pairs from our dataset are (*environment & pollution*, *dirt*, *wastage*), (*border & USA*), (*computer & technology*). We allowed subjects to use multiple phrases to describe the theme to allow for a more diverse set of cliparts to search from when generating the TReAT. We evaluate our TReATs along three dimensions: 1)

Can subjects recognize the word in the TReAT? 2) Can subjects recognize the theme in the TReAT? and 3) Do subjects find the TReAT creative? We conduct independent studies for each of these to eliminate the influence of one on the other.

We compare our approach (**THEME-SOME**) to a version where we replace all letters in the word with cliparts (**THEME-ALL**) to evaluate how replacing a subset of letters affects word recognition (expected to increase) and theme recognition (expected to remain unchanged or even increase because recognizing the word can aid in recognizing the theme), as well as creativity (expected to remain unchanged or even increase because the associated word is more legible as opposed to gibberish). We also compare our approach to an approach that replaces letters with cliparts, but is not constrained by the theme of interest (**NOTHEME-SOME** and **NOTHEME-ALL**). We find the clipart that is closest across all 95⁸ themes in our dataset to replace the letter. This can result in increased word recognition because letters can find a clipart that is more similar (from a larger pool not constrained by the theme), but will result in lower theme recognition accuracy. Note that theme recognition will still likely be higher than chance because the word itself gives cues about the theme. For no-themed clipart, we compare an approach that replaces all letters (i.e., **NOTHEME-ALL**) as well as only a subset of letters (**NOTHEME-SOME**). Finally, as a point of reference, we evaluate a TReAT that simply displays the word in a slightly atypical font (**FONT**). We expect word recognition to be nearly perfect, but theme recognition as well as creativity to be poor. These five different types of TReATs are shown in Figure 6. This gives us a total of 520 TReATs to evaluate (5 types × 104 word-theme input pairs). No AMT workers were repeated across any of these tasks.

Word recognition: We showed each TReAT to 5 subjects on AMT. 461 unique subjects participated in the word recognition study. They were asked to type out the word they see in the TReAT in free-form text. Notice the open-ended nature of the task. Performance of crowd-workers for word recognition of different types of TReATs is shown in Figure 10a. This checks for exact string matching (case-insensitive) between the word entered by subjects and the true word. As a less stringent evaluation, we also compute individual letter recognition accuracy. These were computed only for cases where the length of the word entered by the subject matched the true length of the TReAT because if the lengths do not match, the worker likely made a mistake or was distracted. Letter recognition accuracies are shown in Figure 10b.

As expected, leaving a subset of the letters unchanged leads to a higher recognition rate for **THEME-SOME** and **NOTHEME-SOME** compared to their counterparts, **THEME-ALL** and **NOTHEME-ALL** respectively. Also, **NOTHEME-ALL** and **NOTHEME-SOME** have higher word recognition accuracy than **THEME-ALL** and **THEME-SOME** because the clipart matches are obtained from a larger pool (across all themes rather than from a specific theme). The added signal from the theme of the cliparts in **THEME-ALL** and **THEME-SOME** does not help word

⁸95 since some themes repeat in our 104 (word & theme) pairs.



Figure 6: We evaluate five different approaches for generating TReATs. See text for details.

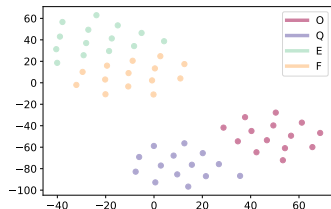


Figure 7: t-SNE plot showing clusters of uppercase O, Q, E and F. Each letter forms its own cluster and visually similar pairs (E & F, O & Q) form super-clusters. However, these super-clusters are far apart from each other due to significant visual differences.

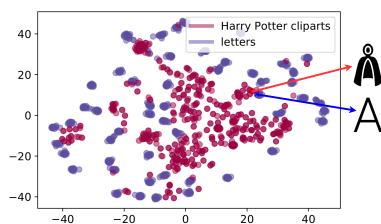


Figure 8: t-SNE plot showing clusters of Harry Potter themed cliparts along with letters. Cliparts which look like A lie close to the cluster of A's in the latent space.

recognition enough to counter this. **NOTHEME-ALL** already has a high recognition rate, leaving little scope for improvement for **NOTHEME-SOME**. Finally, **FONT** has near perfect word recognition accuracy because it contains the word clearly written out. It is not a 100% because of typos on the part of the subjects. In some cases we found that subjects did not read the instructions and wrote out the theme instead of the word itself across all TReATs. These subjects were excluded from our analysis.

Theme recognition: We showed each TReAT to 6 subjects on AMT. 163 unique subjects participated in the theme recognition study. The same theme can be described in many different ways. So unlike word recognition, this task could not be open-ended. For each TReAT, we gave subjects 6 themes as options from which the correct theme is to be identified. These 6 options included the true theme from the 95 themes in our dataset, 2 similar themes, and 3 random themes. The similar themes are the 2 nearest neighbor themes to the true theme in *word2vec* (Mikolov et al. 2013) space. *word2vec* is a popular technique to generate vector representations of a word or “word embeddings” which capture the meaning of the word such that words that share common contexts in language (that is, likely have similar meaning) are located in close proximity to one another in the space. If a theme is described by multiple words, we represent the theme using the average *word2vec* embedding of each word. This is a strategy that is commonly employed in natural language processing to reason about similarities between phrases or even entire sentences (Wieting et al. 2016), (Adi et al. 2017). For example, the options for Figure 4c were 1) home, furniture,

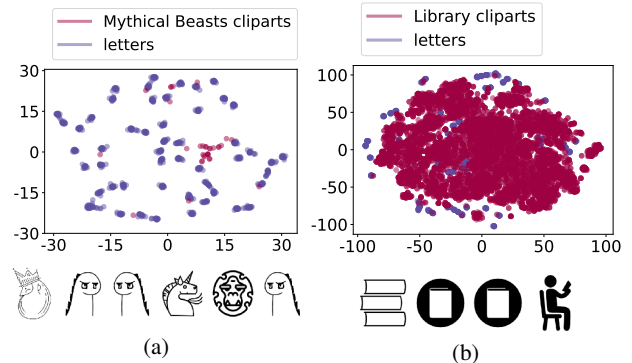


Figure 9: Impact of diversity of cliparts from different themes in the Noun Project on corresponding TReATs. a) TReAT of dragon in theme mythical beast is not legible due to lower coverage of letters by the themed cliparts compared to a TReAT of book in theme library shown in b).

2) drinks, 3) corpse, undertaker, vampire, 4) food, dessert, 5) birds and 6) food with the correct answer being drinks.

We find that 64% of the TReATs were assigned to the correct theme for **THEME-ALL**, 67% for **THEME-SOME**, 43% for **NOTHEME-ALL**, 51% for **NOTHEME-SOME** and 60% for **FONT** respectively. As expected, **NOTHEME-ALL** and **NOTHEME-SOME** have lower theme recognition accuracy than **THEME-ALL** and **THEME-SOME** because **NOTHEME-ALL** and **NOTHEME-SOME** do not use cliparts from specific themes. Notice that theme recognition accuracy is still quite high, because the word itself often gives away cues about the theme (as seen by the theme recognition accuracy of **FONT** that lists the word without any clipart).

This theme recognition rate is a pessimistic estimate because theme options presented to subjects included nearest neighbors to the true theme as distractors. These themes are often synonymous to the true theme. As a less stringent evaluation, we sort the 6 options for each TReAT based on the number of votes the option got across subjects. Figure 10c shows the Mean Reciprocal Rank of the true option in this list (higher is better). We also show Recall@K in Figure 10d that compute how often the true option is in the top-K in this sorted list. Similar trends as described above hold.

Comparing **THEME-SOME** to **THEME-ALL**, we see that replacing only a subset of letters does not hurt theme recognition (in fact, it improves slightly), but improves word recognition significantly. So overall, **THEME-SOME** produces the best TReATs. We see this being played out when TReATs are evaluated for their overall creativity (next). This relates to Schmidhuber’s theory of creativity (Schmidhuber 2010). He argues that data is creative if it exhibits both a learnable or recognizable pattern (and is hence compressible), and novelty. **THEME-SOME** achieves this balance.

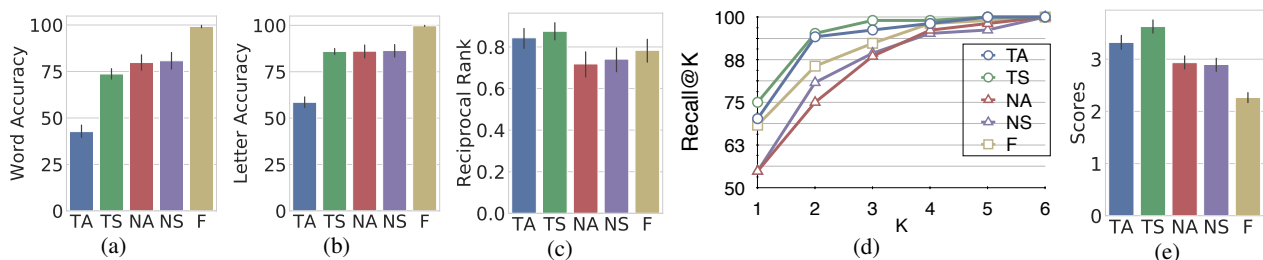


Figure 10: Evaluation of TReATs from five approaches (THEME-ALL (TA), THEME-SOME (TS), NOTHEME-ALL (NA), NOTHEME-SOME (NS), FONT (F)) for a) word recognition; b) letter recognition; c) and d) theme recognition; e) creativity.

Creativity: Recall that our goal here is to create TReATs to depict words with visual elements such that the TReAT leaves an impression on people’s minds. We now attempt to evaluate this. Do subjects find the TReAT intriguing / surprising / fun (i.e., creative)? We showed each TReAT to 5 subjects on AMT. 207 unique subjects participated in the creativity study. They were told: “This is a doodle of [word] in a [theme] theme. On a scale of 1-5, how much do you agree with this statement? This doodle is creative (i.e., surprising and/or intriguing and/or fun). a. Strongly agree (with a grin-like smiley face emoji in green) b. Somewhat agree (with a smiley face in lime green) c. Neutral (with a neutral face in yellow) d. Somewhat disagree (with a slightly frowning face in orange) e. Strongly disagree (with a frowning face in red).” The associated scores were 5 to 1 respectively. Crowd-worker scores are shown in Figure 10e.

THEME-SOME was rated the highest. We believe this is due to a good trade off between legibility and having a theme-relevant depiction that allows for semantic reinforcement. NOTHEME-ALL and NOTHEME-SOME are significantly worse. Recall that they are visual, but not in a theme-specific way. So they are visually interesting, but do not allow for semantic reinforcement. The resultant reduction in creativity is evident. Interestingly, NOTHEME-SOME scores slightly higher than NOTHEME-ALL. This may be because NOTHEME-SOME is not more legible than NOTHEME-ALL (NOTHEME-ALL is already sufficiently legible). With more of the letters visually depicted, NOTHEME-ALL is more interesting. Finally, FONT has a significantly lower creativity score. It is rated lower than neutral, close to the “Somewhat disagree” rating. To get a qualitative sense, we asked subjects to comment on what they think of the TReATs. Some example comments:

THEME-ALL : “cool characters and each one fits the theme of the ocean”, “Its [sic] creative and represents the theme well, but I don’t see disney all that much.”

THEME-SOME : “I like how it uses the image of the US and then a state to spell out the word and looks like something you’d remember.”, “Very fun and intriguing. I like how all the letters are pictures representing a computer mouse.”

NOTHEME-ALL : “It is creative but it has nothing to do with fear.”, “It does a very good job of spelling out CHRISTMAS, but the individual letters are not related to the holiday at all.”

NOTHEME-SOME : “It is somewhat creative, especially the unicorn head for “G”, though I don’t know what any of it has to do with the theme.”, “There are too many icons that

seemingly have nothing to do with the theme.”

FONT : “It just spells out the word, not really a doodle”, “Its not a doodle, its just the word Parrot, so I don’t think its creative at all.”

Future Work

In this section, we discuss some drawbacks of our current model and potential future work.

No Clipart Relevance Score: A comment from a subject evaluating the creativity of Figure 11a (word church & theme pastor, Jesus, people, steeple) was “you need a cross [...] before the general public would [...] get this.” Our approach does not include world knowledge that indicates which symbols are canonical for themes (Noun Project does not provide a relevance score). As a result, our model can not explicitly trade off visual similarity (VS) for theme relevance (TR) – either to compromise on VS to improve TR, or to at least optimize for TR if VS is poor.

No Contextual Querying: Multiple phrases used to describe a theme often lose context when they are individually queried into the Noun Project. For example, the clipart in Figure 11b for (word money & theme finance, banking, support) contains the image of a cheerleader, which is relevant to the phrase support, but is not relevant in the context of the finance theme.

The lack of context also hurts polysemous theme words. bat when used as a keyword with the another keyword bird refers to the creature bat, but in the context of baseball refers to sports equipment.

Imperfect Match Scores: Our automatic similarity score frequently disagrees with our (human) perceptual notion of similarity. E.g., in Figure 11 right, the cliparts used to replace C and N in THEME-ALL look sufficiently similar to the corresponding letters. But the automatic similarity score was low, and so THEME-SOME chose to not replace the letters. Approaches to improve the automatic score can be explored in future work. For instance, in addition to mirror images, using rotated and scaled versions of the cliparts to augment the dataset would help.

Interactive Interface

To mitigate these concerns, we plan to build an interactive tool. Users can choose from the top-*k* clipart matches for each letter. Users can iterate on the input theme descriptions until they are satisfied with the TReAT. Users can also leave

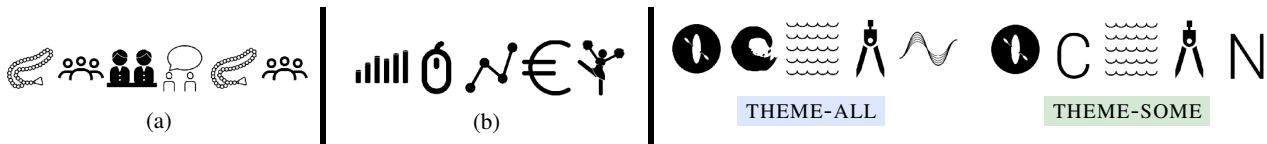


Figure 11: Example failure modes of our approach – a) more theme-relevant icons such as the cross should be used to depict the theme *pastor*, *Jesus*, *people*, *steeple*; b) lack of context wherein *support* here actually refers to *financial* support, and not the motivational support which comes from a cheerleader depicting *y* in the word *money*; Right: C and N are replaced in our final **THEME-SOME** TReAT even when the matches were actually quite relevant and visually fitting.

the theme unspecified in which case we can use the word itself as the theme. Finally, users can choose which letters to replace in **THEME-SOME** like TReATs.

Conclusion

In this work, we introduce a computational approach for semantic reinforcement called TReAT – Thematic Reinforcement for Artistic Typography. Given an input word and a theme, our model generates a “doodle” (TReAT) for that word using cliparts associated with that theme. We evaluate our TReATs for word recognition (can a subject recognize the word being depicted?), theme recognition (can a subject recognize what theme is being illustrated in the TReAT?), and creativity (overall, do subjects find the TReATs surprising / intriguing / fun?). We find that subjects can recognize the word in our TReATs 74% of the time, can recognize the theme 67% of the time, and on average “Somewhat agree” that our TReATs are creative.

References

- [Adi et al. 2017] Adi, Y.; Kermany, E.; Belinkov, Y.; Lavi, O.; and Goldberg, Y. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *ICLR*. 6
- [Atarsaikhan et al. 2017] Atarsaikhan, G.; Iwana, B. K.; Narusawa, A.; Yanai, K.; and Uchida, S. 2017. Neural font style transfer. In *ICDAR*. 2
- [Atarsaikhan, Iwana, and Uchida 2018] Atarsaikhan, G.; Iwana, B. K.; and Uchida, S. 2018. Contained neural style transfer for decorated logo generation. In *IAPR DAS*. 2
- [Azadi et al. 2018] Azadi, S.; Fisher, M.; Kim, V. G.; Wang, Z.; Shechtman, E.; and Darrell, T. 2018. Multi-content gan for few-shot font style transfer. In *CVPR*. 2
- [Ballard 1987] Ballard, D. H. 1987. Modular learning in neural networks. In *AAAI*. 3
- [Campbell and Kautz 2014] Campbell, N. D., and Kautz, J. 2014. Learning a manifold of fonts. *Trans. on Graphics*. 2
- [Clawson et al. 2012] Clawson, T. H.; Leafman, J.; Nehrenz Sr, G. M.; and Kimmer, S. 2012. Using pictograms for communication. *Military medicine*. 2
- [Cunha et al. 2018] Cunha, M.; Martins, P.; João; and Machado, P. 2018. How shell and horn make a unicorn: Experimenting with visual blending in emoji. In *ICCC*. 2
- [Frutiger 1989] Frutiger, A. 1989. Signs and symbols. *Their design and meaning*. 2
- [Ha and Eck 2018] Ha, D., and Eck, D. 2018. A neural representation of sketch drawings. In *ICLR*. 2
- [Isola et al. 2017] Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2017. Image-to-image translation with conditional adversarial networks. In *CVPR*. 2
- [Krizhevsky, Sutskever, and Hinton 2012] Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*. 4
- [Martins et al. 2015] Martins, P.; Urbancic, T.; Pollak, S.; Lavrac, N.; and Cardoso, A. 2015. The good, the bad, and the aha! blends. In *ICCC*. 2
- [Martins et al. 2018] Martins, T.; Correia, J.; Costa, E.; and Machado, P. 2018. Evotype: towards the evolution of type stencils. In *International Conference on Computational Intelligence in Music, Sound, Art and Design*. 2
- [Mikolov et al. 2013] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*. 6
- [Mirza and Osindero 2014] Mirza, M., and Osindero, S. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*. 2
- [Rawlinson 2007] Rawlinson, G. 2007. The significance of letter position in word recognition. *IEEE Aerospace and Electronic Systems Magazine*. 4
- [Sage et al. 2018] Sage, A.; Agustsson, E.; Timofte, R.; and Van Gool, L. 2018. Logo synthesis and manipulation with clustered generative adversarial networks. In *CVPR*. 2
- [Schmandt-Besserat 2015] Schmandt-Besserat, D. 2015. The evolution of writing. *International Encyclopedia of the Social and Behavioral Sciences: Second Edition*. 2
- [Schmidhuber 2010] Schmidhuber, J. 2010. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*. 6
- [Shiojiri and Nakatani 2013] Shiojiri, M., and Nakatani, Y. 2013. Visual language communication system with multiple pictograms converted from weblog texts. In *IASDR*. 2
- [Takasaki and Mori 2007] Takasaki, T., and Mori, Y. 2007. Design and development of a pictogram communication system for children around the world. In *Intercultural collaboration*. 2
- [Wieting et al. 2016] Wieting, J.; Bansal, M.; Gimpel, K.; and Livescu, K. 2016. Towards universal paraphrastic sentence embeddings. In *ICLR*. 6
- [Xiao and Linkola 2015] Xiao, P., and Linkola, S. 2015. Vis-mantic: Meaning-making with images. In *ICCC*. 2
- [Zhu et al. 2017] Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*. 2

Beyond Imitation: Generative and Variational Choreography via Machine Learning

Mariel Pettee
Department of Physics
Yale University
New Haven, CT 06511
mariel.pettee@yale.edu

Chase Shimmin
Department of Physics
Yale University
New Haven, CT 06511
chase.shimmin@yale.edu

Douglas Duhaime
Digital Humanities Laboratory
Yale University
New Haven, CT 06511
douglas.duhaime@yale.edu

Ilya Vidrin
Coventry University
Centre for Dance Research
Harvard University Dance Program
Cambridge, MA 02138
ilya_vidrin@mail.harvard.edu

Abstract

Our team of dance artists, physicists, and machine learning researchers has collectively developed several original, configurable machine-learning tools to generate novel sequences of choreography as well as tunable variations on input choreographic sequences. We use recurrent neural network and autoencoder architectures from a training dataset of movements captured as 53 three-dimensional points at each timestep. Sample animations of generated sequences and an interactive version of our model can be found at <http://www.beyondimitation.com>.

Introduction

“I didn’t want to imitate anybody. Any movement I knew, I didn’t want to use.” (Jennings 2009) Eminent postmodern dance choreographer Pina Bausch felt the same ache that has pierced artists of all generations – the desire to generate something truly original from within the constraints of your own body.

Recent technologies enabling the 3D capture of human motion as well as the analysis and prediction of timeseries datasets with machine learning have opened provocative new possibilities in the domain of movement generation. In this paper, we introduce a suite of configurable machine learning tools to augment a choreographer’s workflow.

Many generative movement models from recent publications use Recurrent Neural Networks (RNNs) (Graves 2013) as their fundamental architecture (McCormick et al. 2015; Crnkovic-Friis and Crnkovic-Friis 2016; Alemi, Franoise, and Pasquier 2017; Li et al. 2017; James 2018; Marković and Malešević 2018). Others create methods to draw trajectories through a lower-dimensional space of possible human poses constructed through techniques such as Kernel Principal Component Analysis (KPCA) (Berman and James 2015; Schlkopf, Smola, and Miller 1998). We build upon existing RNN techniques with higher-dimensional datasets and introduce autoencoders (Kingma and Welling 2013) of both poses and sequences of poses to construct variations on input sequences of movement data and novel unprompted sequences sampled from a lower-dimensional latent space.

Our models not only generate new movements and dance sequences both with and without a movement prompt, but can also create infinitely many variations on a given input

phrase. These methods have been developed using a dataset of improvisational dance from one of the authors herself, recorded using a state-of-the-art motion capture system with a rich density of datapoints representing the human form. With this toolset, we equip artists and movement creators with strategies to tackle the challenge Bausch faced in her own work: generating truly novel movements with both structure and aesthetic meaning.

Context within Dance Scholarship

Dance scholarship, psychology, and philosophy of the past century has increasingly seen movement as embodied thought. Prominent proposals including psychologist Jean Piaget’s sensorimotor stage of psychological development, the philosopher Maurice Merleau-Ponty’s “phenomenology of embodiment”, and Edward Warburton’s concept of *dance enaction* have guided us today to view the human body as an essential influencer of cognition and perception (Warburton 2011).

Our team’s vision for the future of creative artificial intelligence necessitates the modeling of not only written, visual, and musical thought, but also kinesthetic comprehension. In light of the modern understanding of movement as an intellectual discipline, the application of machine learning to movement research serves not as a mere outsourcing of physical creative expressiveness to machines, but rather as a tool to spark introspection and exploration of our embodied knowledge.

Concurrently with this branch of research, choreographers have wrestled with the problem of constructing a universal language of movement. Movement writing systems currently in use today such as Labanotation, Benesh Choreology, and Eshkol-Wachmann Notation can be effective in limited use cases, but make culturally-specific assumptions about how human bodies and types of motion should be abstracted and codified (Farnell 1996).

It is not our aim to replace existing methods of dance notation. However, we note the significance of 3D motion-capture techniques and abstract latent spaces in potentially reorienting movement notation away from culturally-centered opinions such as qualities of movement or which segments of the body get to define movement. Rather than gravitating in the direction of defining “universal” movement signifiers, we see this work as more aligned with the expressive figures

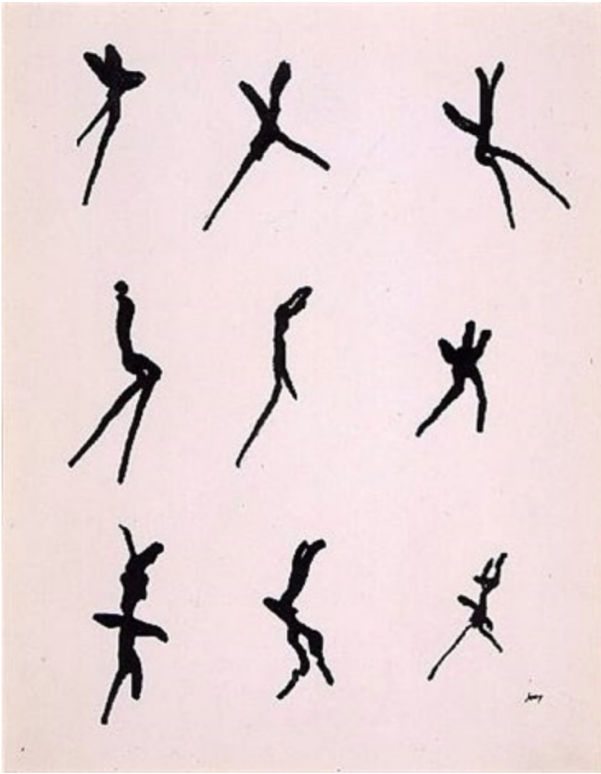


Figure 1: Henri Michaux’s notion of *envie cinétique*, or “kinetic desire”, is represented by expressive, personal, and idiosyncratic gestures in calligraphic ink in his series *Mouvements* (Noland 2009).

generated by the visual artist Henri Michaux in an attempt to capture what he called *envie cinétique*, or “kinetic desire” – in other words, the pure impulse to move (see Figure 1). We therefore avoid limiting our generated movement outputs to only physically-achievable gestures, as this would only serve to limit the potential imaginative sparks lying dormant in these sequences.

Ethics in the philosophy of emerging media raise particular questions about how technology impacts what it means to be human, especially given the way constraints and resources of technology affect our embodied dispositions. When we consider the ethical dimensions of choreography in the context of machine learning, one major benefit is the opportunity to reflect on habits by observing, interpreting, and evaluating what is generated technologically. The normative problems that ensue are manifold: if we ascribe great value to what we see, we may find ourselves in a position where we envy an algorithm’s capacity to generate novel choreography. This may in turn lead us to cast judgement on ourselves and doubt our own human-created choreographies.

While technology may provide new insights into patterns within dance sequences, it also inevitably leads to normative discussion about what it means to choreograph well, or appropriately, or even creatively. This opens the door for replacing

our own practice with algorithms that could ostensibly rob us of the opportunity to get better at choreography, or learn to be more creative. While this may seem a bit like catastrophizing, these normative problems can lead to real ethical concerns related not only to artistic practice, but to education more broadly.

Several prominent choreographers have sought out both motion capture and machine learning tools to augment their practice, from Bill T. Jones and the OpenEndedGroup’s 1999 motion capture piece *Ghostcatching* to William Forsythe to Merce Cunningham (Bill T. Jones and Eshkar 1999; Naugle 2000; Marc Downie and Eshkar 2001). Wayne McGregor recently collaborated with Google Arts & Culture to create *Living Archive*, a machine learning-based platform to generate a set of movements given an input sequence derived from a video, although details of the technical implementation of this project are not yet publicly released (LePrince-Ringuet 2018).

Our work represents a unique direction in the space of “AI-generated” choreographies, both computationally and artistically. Computationally, we combine high-dimensional and robust 3D motion capture data with existing RNN-based architectures as well as introducing the use of autoencoders for 3D pose and movement sequence generation. Artistically, we deviate from having novel predicted sequences as the only end goal – in addition to this functionality, we grant choreographers the power to finely-tune existing movement sequences to find subtle (or not-so-subtle) variations from their original ideas.

Methods

Training data was recorded in a studio equipped with 20 Vicon Vantage motion-capture cameras and processed with Vicon Shogun software. This data consists of the positions of 53 fixed vertices on a dancer in 3 dimensions through a series of nearly 60,000 temporal frames recorded at 35 fps, comprising approximately 30 minutes of real-time movement. Each frame of the dataset is transformed such that the overall average (x,y) position per frame is centered at the same point and scaled such that all of the coordinates fit within the unit cube. The primary author, who has an extensive background in contemporary dance, supplied the training data. The data was then exported to Numpy array format for visualization and processing in Python, and to JSON format for visualization with the interactive 3D Javascript library `three.js`. The neural network models were constructed using Keras with a Tensorflow backend.

In the following subsections, we describe two methods for generating dance movement in both conditional (where a prompt sequence of fixed length is provided) and unconditional (where output is generated without input) modes. The first method involves a standard approach to supervised training for sequence generation: an RNN is presented with a sequence of training inputs, and is trained to predict the next frame(s) in the sequence. The second method takes advantage of autoencoders to convert either an arbitrary-length sequence of dance movement into a trajectory of points in a low-dimensional latent space, or a fixed-length sequence to a single point in a higher-dimensional latent space.

LSTM+MDN

The model proposed in *chor-rnn* (Crnkovic-Friis and Crnkovic-Friis 2016) uses RNNs to generate dance from a dataset of 25 vertices captured with a single Kinect device, which requires the dancer to remain mostly front-facing in order to capture accurate full-body data. Our RNN model uses an input layer of size $(53 \times 3 \times m)$ to represent 53 three-dimensional vertices with no rotational restrictions in a prompt sequence of m frames at a time. These sequences are then input to a series of LSTM layers, typically three, followed by a Mixture Density Network (Aleml, Franoise, and Pasquier 2017) (see Appendix A) which models proposals for the vertex coordinates of the subsequent n frames. The LSTM layers ensure the model is capable of capturing long-term temporal dependencies in the training data, while the MDN layer ensures generated sequences are dynamic and do not stagnate on the conditional average of previous vertex sequences (Bishop 1994). The network is trained using supervised pairs of sequences by minimizing the negative log likelihood (NLL) of the proposed mixture model.

We also developed a modification of this structure using Principal Component Analysis (PCA) to reduce the dimensionality of the input sequences. This reduces the amount of information that must be represented by each LSTM layer. We then invert the PCA transformation to convert generated sequences in the reduced-dimensional space back into the $(53 \times 3 \times n)$ -dimensional space.

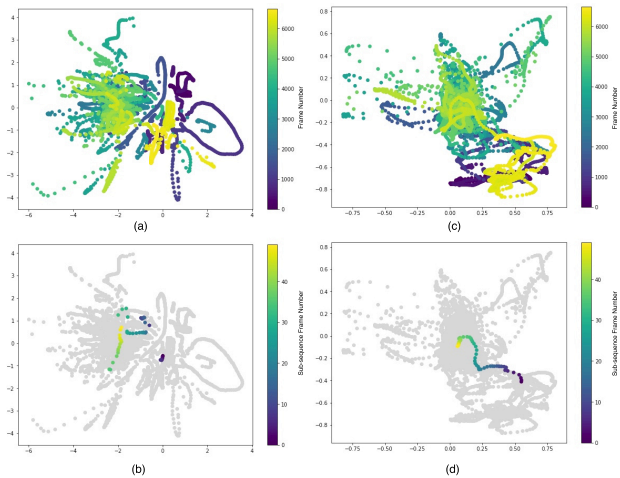


Figure 2: **(a)** The 2-dimensional latent space of an autoencoder trained on a subset of the full dataset. The frame numbers show the procession of the sequence through time at a frame rate of 35 fps. **(b)** An example sequence of real training data is highlighted in this latent space. Note that its structure is highly noncontinuous. **(c)** The 2-dimensional latent space of an autoencoder trained on the same subset of data as the previous plots, but with the angular orientation of the frames subtracted. **(d)** The same sequence of real training data is highlighted, showing a much smoother and more continuous structure.

Autoencoder Methods

Unlike the RNN methods described above, autoencoders can learn features of the training data with a less directly supervised approach. The input and output layers are identical in dimensionality, while the intermediate layer or layers are of a reduced dimension, creating a characteristic bottleneck shape in the network architecture. The full network is then trained to replicate the training samples as much as possible by minimizing the mean-squared error loss between the input and the generated output. The network therefore learns a reduced dimensionality representation of “interesting” features in an unsupervised manner, which can be exploited in the synthesis of new types of movement.

While a well-trained autoencoder merely mimics any input data fed into it, the resulting network produces two useful artifacts: an *encoder* that maps inputs of dimension $(53 \times 3 \times m)$ to a $(d \times m)$ -dimensional space ($d < 159$) and a *decoder* that maps $(d \times m)$ -dimensional data back into the original dimensionality of $(53 \times 3 \times m)$. This allows us to generate new poses and sequences of poses by tracing paths throughout the $(d \times m)$ -dimensional latent space which differ from those found in the training data.

While there are many other dimensional reduction techniques for data visualization, such as PCA, UMAP, and t-SNE (Pearson 1901; McInnes, Healy, and Melville 2018; van der Maaten and Hinton 2008), a significant advantage of autoencoders is that they learn a nonlinear mapping to the latent space that is by construction (approximately) invertible. Some differences between these other dimensionality-reducing techniques are illustrated in Figure 3.

In principle, autoencoders can be used to synthesize new dance sequences by decoding any arbitrary trajectory through the latent space. We prioritize continuity and smoothness of paths in the latent space when possible, as this allows human-generated abstract trajectories (for example, traced on a phone or with a computer mouse) a greater likelihood of creating meaningful choreographies. These qualities of trajectories in the latent space are most prevalent in PCA and our autoencoder methods (see Figure 3). However, as PCA is a linear dimensionality-reduction method, it is far more limited in ability to conform to the full complexity of the realistic data manifold compared to autoencoder methods.

The autoencoders’ latent spaces do tend to produce mostly continuous trajectories for real sequences in the input data. This continuity can be greatly enhanced by subtracting out angular and positional orientation of the dancer, as shown in Figure 2. Removing these dimensions of variation further reduces the amount of information that must be stored by the autoencoder and allows it to create less convoluted mappings of similar poses regardless of the overall spatial orientation of the dancer.

However, absent a deliberate human-selected trajectory as an input, it is *a priori* unclear how to select a meaningful trajectory, i.e., one that corresponds to an aesthetically or artistically interesting synthetic performance.

In order to address this limitation, and to give some insight into the space of “interesting” trajectories in the latent space, we take another approach in which a second autoencoder is trained to reconstruct fixed-length sequences of

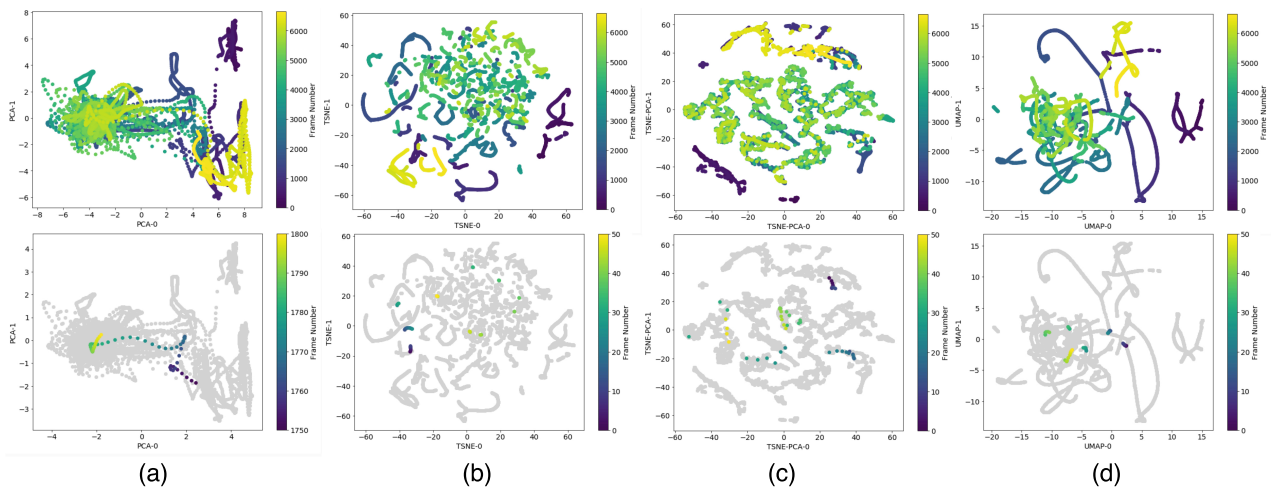


Figure 3: A variety of 2D latent spaces are compared across multiple linear and nonlinear dimensionality-reduction techniques (excluding autoencoders): **(a)** PCA, **(b)** t-SNE, **(c)** t-SNE following PCA, and **(d)** UMAP. The top row shows full latent spaces for a subset of the training data, while the bottom row highlights the same example sequence of 50 frames in each space. All but PCA show a very segmented and discontinuous path for the sequence across the latent space. Our autoencoder techniques (see Figure 2) are comparable to PCA in terms of continuity of the paths in latent space, but have a much higher capacity to learn complex, nonlinear relationships than PCA alone.

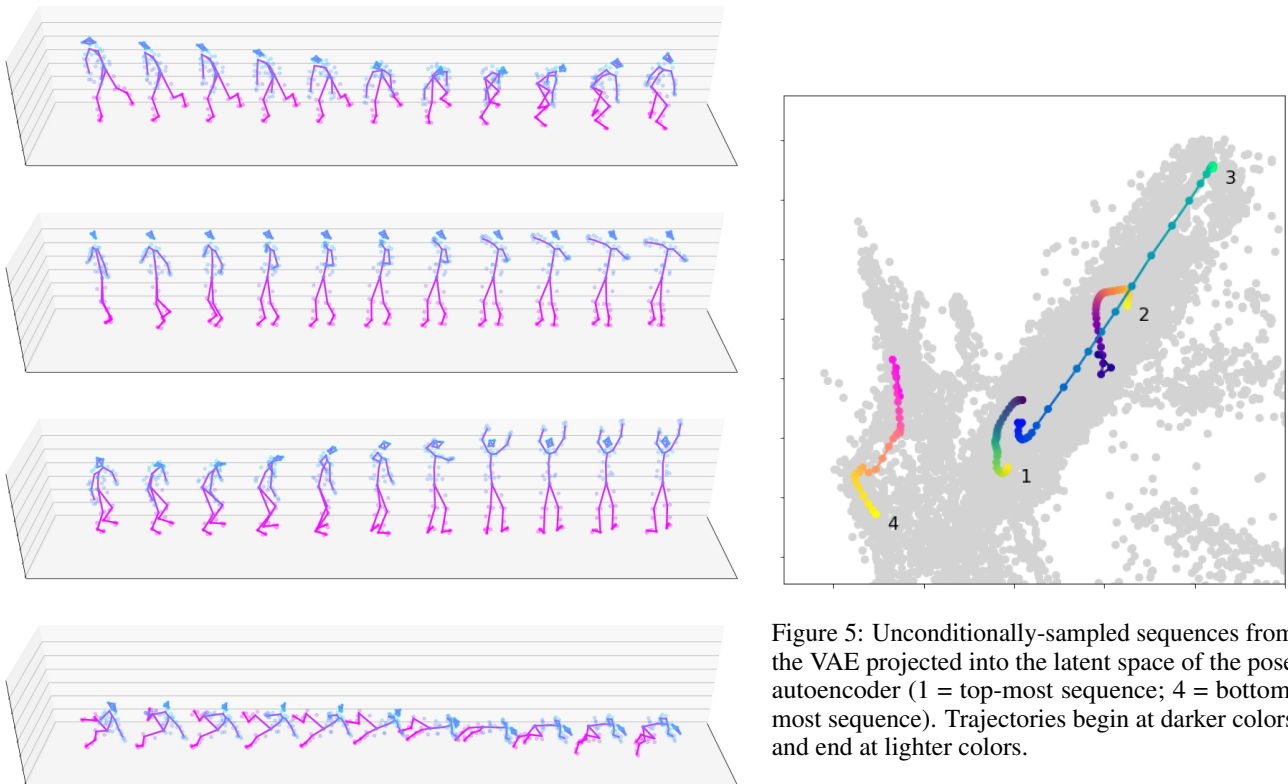


Figure 4: Unconditionally-sampled sequences from the VAE.

Figure 5: Unconditionally-sampled sequences from the VAE projected into the latent space of the pose autoencoder (1 = top-most sequence; 4 = bottom-most sequence). Trajectories begin at darker colors and end at lighter colors.

dance poses by mapping each sequence to a single point in a high-dimensional latent space. Moreover, we train this network as a Variational Autoencoder (VAE) (Larsen, Sønderby, and Winther 2015) which attempts to learn a latent space whose distribution is compatible with a $(d \times m)$ -dimensional Gaussian. Sampling from this latent space results in unconditionally-generated sequences that are realistic and inventive (see Figure 4). For each sampling, we look at a single point in the latent space corresponding to a fixed-length movement sequence. Within the scope of this paper, we do not attempt to impose any continuity requirements from one sampling to the next. Latent space points are chosen approximately isotropically. This creates a complementary creative tool to our previously-described traditional autoencoder for poses. We anticipate that choreographers and researchers could draw continuous paths through the latent space of poses to generate new movements as well as sample from the VAE latent space to generate new movement phrases and/or variations on existing phrases.

With both standard and variational autoencoders trained to replicate single poses and sequences of poses respectively, we introduce some techniques for taking a given input phrase of movement and generating infinitely many variations on that phrase. We define “variation” to mean that the overall spirit of the movement be preserved, but implemented with slightly different timing, intensity, or stylistic quality.

After identifying a desired dance phrase from which to create variations, we identify the sequence of points in the latent space representing that sequence of poses. We first constructed trajectories close to the original sequence by adding small sinusoidal perturbations to the original sequence. This created sequences resembling the original phrase, but with an oscillatory frequency that was apparent in the output. This frequency could be tuned to the choreographer’s desired setting, if the oscillatory effect is desired. However, we also sought out a method that constructed these paths in a less contrived manner.

For a VAE trained on sequences of poses, each point in the latent space represents an entire sequence of a fixed length m . We can construct variations on the input sequence by adding a small amount of random noise to the latent point and then applying the decoder to this new point in the latent space. This creates a new generated variation on the original sequence, with a level of “originality” that scales with the amount of noise added. Since the VAE’s latent space has been constrained to resemble a Gaussian distribution, we can sample frequently from the latent space within several standard deviations of the origin without observing highly unphysical output sequences. Sampling within less than approximately 0.5σ tends to give very subtle variations, usually in timing or expressiveness in the phrase. Sampling within approximately 1 to 2σ gives more inventive variations that deviate further from the original while often preserving some element of the original, e.g. a quick movement upwards of a limb or an overall rotational motion. Sampling within 3 to 4σ and higher can produce myriad results ranging from no motion at all to extreme warping of the body to completely destroying the sense of a recognizable human shape.

The relationship between these two latent spaces – that of

the pose autoencoder and that of the sequence VAE – may be exploited to gain insight into the variations themselves. Points in the VAE latent space directly map to trajectories in the pose autoencoder space. By introducing a slight amount of noise to the point in the VAE latent space corresponding to a desired input sequence, we may decode nearby points to construct trajectories in pose space that are highly related to the original input sequence. Examples of variations from reference sequences are shown in Figures 6 - 11.

Results and Discussion

Both the RNN+MDN and autoencoded outputs created smooth and authentic-looking movements. Animations of input and output sequences for various combinations of our model parameters may be viewed here: <http://www.beyondimitation.com>.

Training the RNN+MDN with a PCA dimensionality reduction tended to improve the quality of the generated outputs, at least in terms of the reconstruction of a realistic human body. We used PCA to transform the input dataset into a lower-dimensional format that explains 95% of its variance. This transformation of the training data shortened the training time for each epoch by up to 15%, though test accuracy was not significantly affected. The output resulted in a realistic human form earlier in the training than without the application of PCA. In the future, we may also investigate nonlinear forms of dimensionality reduction to further improve this technique.

The architectures used for the RNN+MDN models included 3 LSTM layers with sizes varying from 32 to 512 nodes. They took input sequences of length m ranging from 10 to 128 and predicted the following n frames ranging from 1 to 4 with a learning rate of 0.00001 and the Adam optimizer (Kingma and Ba 2014).

The final architecture for the pose autoencoder comprises an encoder and a decoder each with two layers of 64 nodes with LeakyReLU activation functions with $\alpha = 0.2$ and compiled with the Adam optimizer. The pose autoencoder takes inputs of shape (53×3) and maps them into a latent space of 32 dimensions. Training this over 80% of our full dataset with a batch size of 128 and a learning rate of 0.0001 produced nearly-identical reconstructions of frames from the remaining 20% of our data after about 50 epochs. We also trained a modification of this architecture with a data augmentation technique that added random offsets between $[0, 1]$ to the \hat{x} and \hat{y} axes. This did not yield a significant advantage in terms of test accuracy, however, so we did not use it for our latent space explorations.

The final architecture for the sequence VAE also comprises an encoder and a decoder, each with 3 LSTM layers with 384 nodes and 1 dense layer with 256 nodes and a ReLU activation function, where 256 represents the dimensionality of the latent space. The model was compiled with the Adam optimizer. The VAE maps inputs of shape $(53 \times 3 \times l)$, where l is the fixed length of the movement sequence, to the $(256 \times l)$ -dimensional latent space and then back to their original dimensionality. We used input sequences of length $l = 128$, which corresponds to about 4 seconds of continuous movement. We augmented our data by rotating the frames

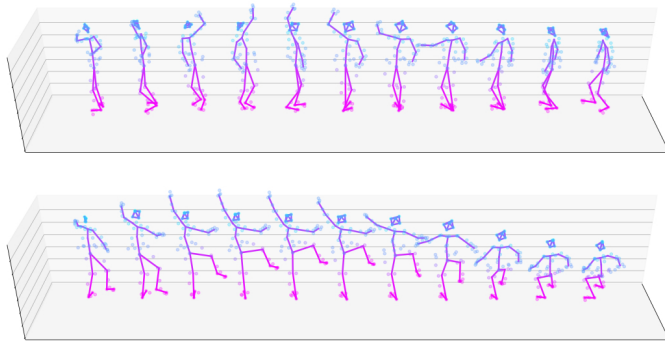


Figure 6: A reference input sequence (above) and a generated variation sequence (below) with 0.5σ noise added to the input's representation in latent space, both with lengths of 32 frames (time progressing from left to right). While the reference sequence includes a rotation, the generated variation removes the spin, while the movements of the left arm are synchronous in both cases.

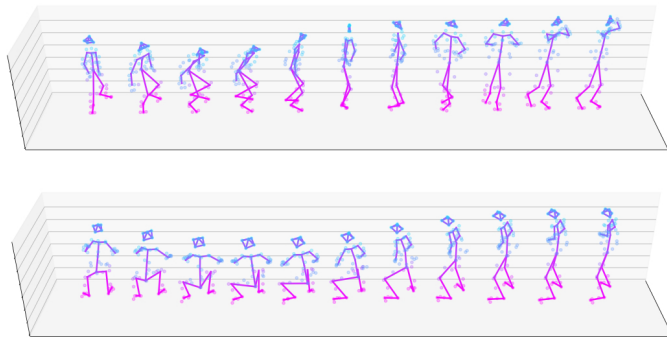


Figure 8: A reference input sequence (above) and a generated variation sequence (below) with 0.5σ noise added to the input's representation in latent space, both with lengths of 32 frames (time progressing from left to right). The generated variation preserves the rising motion but adds a rotation.

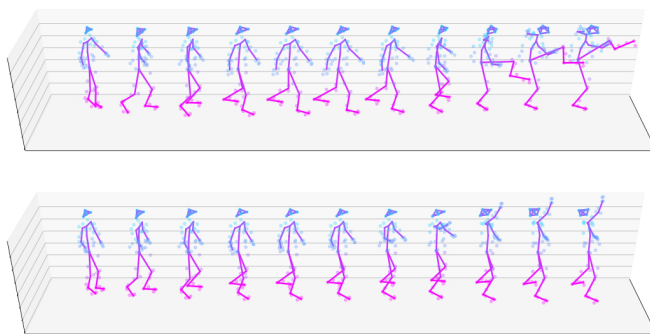


Figure 10: A reference input sequence (above) and a generated variation sequence (below) with 0.5σ noise added to the input's representation in latent space, both with lengths of 32 frames (time progressing from left to right). The reference sequence features a kick, while the variation instead translates this upward motion into the arms, rather than the feet.

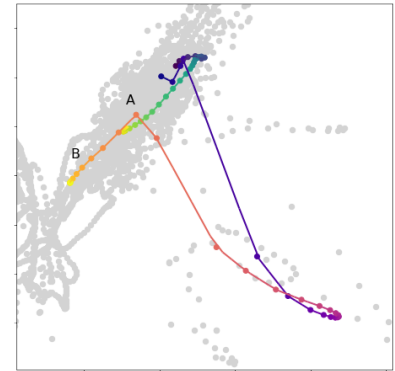


Figure 7: Reference (A) and generated (B) variation sequences projected into the pose autoencoder space. Trajectory colors go from dark to light over time.

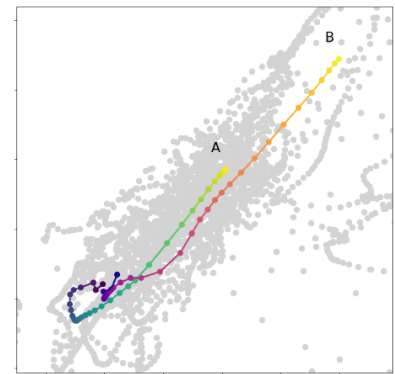


Figure 9: Reference (A) and generated (B) variation sequences projected into the pose autoencoder space. Trajectory colors go from dark to light over time.

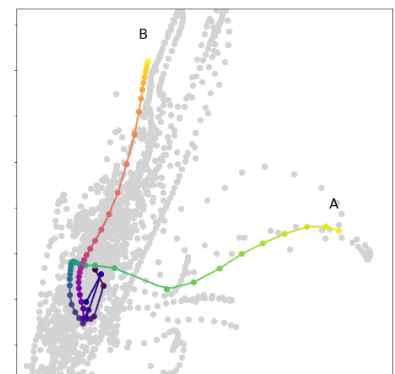


Figure 11: Reference (A) and generated (B) variation sequences projected into the pose autoencoder space. Trajectory colors go from dark to light over time.

in each batch by a randomly-chosen $\theta \in [0, 2\pi]$. The VAE was trained with a learning rate of 0.001, a Kullback-Leibler weight = 0.0001, and a Mean Squared Error (MSE) loss scaled by the approximate resolution of the motion capture data for about 1 day on a CuDNN-enabled GPU.

Sampling from the latent space of standard and variational autoencoders for both poses and sequences provided a rich playground of generative movements. We are particularly interested in the dynamic range provided by these tools to create variations on input sequences: by increasing the magnitude of the perturbation of the latent sequence to be decoded, choreographers can decide how ‘creative’ the outputs should look. By opting for either a standard or variational autoencoder, choreographers can sample from latent spaces with a bit more or a bit less similarity in the movements themselves to the training data. Adding sinusoidal perturbations as well as generating stylistically-related variations by exploiting the relationship between these two latent spaces proved effective and compelling methods for creating choreographic variations. The subtlety and smoothness with which we can vary input sequences using the VAE also underscores that the model is truly generating new outputs rather than memorizing the input data.

These methods have already been effective at sparking choreographic innovation in the studio. They center the choreographer’s embodied knowledge as something to be modeled and investigated – not just as a compendium of possible bodily positions, but as a complex and high-dimensional landscape from which to sample movements both familiar and foreign. Movements throughout these abstract landscapes can be constructed in a variety of ways depending on the application. For example:

- For a choreographer seeking primarily to document their practice, training these models allows them to save not only the physical motions captured in the motion capture data, but also their *potential* for movement creation as approximated by a well-trained model. Different models may be saved from various periods of their practice and compared or re-explored indefinitely.
- For a choreographer looking to construct a new piece out of their own typical patterns of movement, sampling from within 1σ in the VAE latent space can generate multiple natural-looking phrases that can then be stitched together in the studio to create a cohesive piece. They could also prompt new sequences of arbitrary length following from existing choreography via the RNN+MDN model.
- For a choreographer who wants to understand and perhaps break out of their typical movement patterns, analyzing the latent space of the pose autoencoder can be instructive. Visualizing trajectories through the space can inform what areas lie unexplored. Drawing continuous paths through the latent space can then construct new phrases that might otherwise never emerge from an improvisation session.
- A choreographer might also use these methods to support teaching movements to others. By comparing trajectories in the same latent space, students can track their mastery of a given movement sequence.

These creative tools allow a mode of documentation that opens up valuable reflection in the recursive process of movement creation. Since a significant portion of the choreographic process can be kinesthetically driven, it is useful to be able to externalize movement into the visual domain in order to reflect on the architecture and design of the choreography. This resource may double as a limitation if dance-makers rely solely on the visual aspect of choreography. Just as a mirror can serve as a double-edged sword in dance practice, these tools make explicit the possibilities of differentiation between the internal, kinesthetic dimension of movement research and the external, visual one.

Generating novel movement allows us to see potential flow of choreographic patterns, which makes negotiating the aesthetic dimension richer if we take the time to evaluate why something looks subjectively unnatural. In this way, a dance-maker has a chance to articulate more clearly their own aesthetic preferences for choreographic intention and execution.

As our title suggests, “beyond imitation” also points to the important distinction between creative expression and research-based inquiry. While dance-making certainly involves generation that is spontaneous and intuitive, choreographers may also take years honing and developing sequences that are deeply textured and multi-faceted. Disrupting any implicit hierarchies, these tools enable documentation of the systematic, recursive process of dance-making that is often so invisible and mysterious.

Future technical work to develop these methods will include the investigation of nonlinear, invertible data-reduction techniques as a form of pre-processing our inputs, other neural network-based models designed to work with timeseries data such as Temporal Convolutional Networks, and more sophisticated methods for sampling from latent spaces.

We can also increase the size of our training dataset by sourcing data not only from motion capture sessions, but also using OpenPose software to extract pose information from dance videos or even a laptop camera (Hidalgo and others 2018; Jones 2019). This could open up a provocative path in machine-augmented choreography: generating movements in the styles of any number of prominent choreographers.

Feedback from other choreographers who used our interactive models also indicated that it would be interesting to extend our current dataset with additional data focused on the isolation of certain regions of the body and/or modalities of movement. Our next steps in extending this work will also include exploring latent spaces of multiple dancers. While only solo dances were captured for the studies in this paper, the Vicon system can readily accommodate multiple simultaneous dancers, which will allow us to explore the generation of duets and group choreographies.

Acknowledgements

We would like to thank Raymond Pinto for many useful conversations and insights as well as providing additional movement data for future studies. This work has been supported by the generosity of the Yale Center for Collaborative Arts & Media, the Yale Womens’ Faculty Forum, and the Yale Digital Humanities Lab.

Appendix A: Mixture Density Networks

The structure of a Mixture Density Network, as laid out in detail in (Bishop 1994), allows us to sample our target predictions from a linear combination of m Gaussian distributions, each multiplied by an overall factor of α_i , rather than from a single Gaussian. The probability density is therefore represented by

$$p(\vec{t} | \vec{x}) = \sum_{i=1}^m \alpha_i(\vec{x}) \phi_i(\vec{t} | \vec{x})$$

where \vec{x} represents our input data, \vec{t} represents a given predicted output, m represents the total number of Gaussian distributions in the mixture, and c represents the total number of components to predict (here, 53×3 for each timeslice). Each of the Gaussian distributions is modeled as:

$$\phi_i(\vec{t} | \vec{x}) = \frac{1}{(2\pi)^{\frac{c}{2}} \sigma_i(\vec{x})^c} e^{-\frac{|\vec{t} - \vec{\mu}_i(\vec{x})|^2}{2\sigma_i(\vec{x})^2}}$$

Here, $\vec{\mu}_i(\vec{x})$ and $\sigma_i(\vec{x})$ represent the mean values and variances for each component of the generated output.

References

- Alemi, O.; Franoise, J.; and Pasquier, P. 2017. GrooveNet: Real-time music-driven dance movement generation using artificial neural networks. https://www.researchgate.net/publication/318470738_GrooveNet_Real-Time_Music-Driven_Dance_Movement_Generation_using_Artificial_Neural_Networks.
- Berman, A., and James, V. 2015. Kinetic imaginations: Exploring the possibilities of combining AI and dance. In *Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI'15*, 2431–2437. AAAI Press.
- Bill T. Jones, P. K., and Eshkar, S. 1999. Ghostcatching. <http://openendedgroup.com/artworks/gc.html>.
- Bishop, C. M. 1994. Mixture density networks. *Neural Computing Research Group Report*.
- Crnkovic-Friis, L., and Crnkovic-Friis, L. 2016. Generative choreography using deep learning. <https://arxiv.org/abs/1605.06921>.
- Farnell, B. 1996. Movement writing systems. *The World's Writing Systems*.
- Graves, A. 2013. Generating sequences with recurrent neural networks. <https://arxiv.org/abs/1308.0850>.
- Hidalgo, G., et al. 2018. OpenPose. <https://github.com/CMU-Perceptual-Computing-Lab/openpose>.
- James, A. B. . V. 2018. Learning as performance: Autoencoding and generating dance movements in real time. *Computational Intelligence in Music, Sound, Art and Design* 256–266.
- Jennings, L. 2009. Obituary: Pina Bausch. <https://www.theguardian.com/stage/2009/jul/01/pina-bausch-obituary-dance>.
- Jones, B. T. 2019. Body movement language: AI sketches with Bill T. Jones. <https://billtjonesai.com/>.
- Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization.
- Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.
- Larsen, A. B. L.; Sønderby, S. K.; and Winther, O. 2015. Autoencoding beyond pixels using a learned similarity metric. *CoRR* abs/1512.09300.
- LePrince-Ringuet, D. 2018. Google's latest experiment teaches AI to dance like a human. <https://www.wired.co.uk/article/google-ai-wayne-mcgregor-dance-choreography>.
- Li, Z.; Zhou, Y.; Xiao, S.; He, C.; and Li, H. 2017. Auto-conditioned LSTM network for extended complex human motion synthesis. *CoRR* abs/1707.05363.
- Marc Downie, P. K., and Eshkar, S. 2001. Loops. <http://openendedgroup.com/artworks/loops.html>.
- Marković, D., and Malešević, N. 2018. Adaptive interface for mapping body movements to sounds. In Liapis, A.; Romero Cardalda, J. J.; and Ekárt, A., eds., *Computational Intelligence in Music, Sound, Art and Design*, 194–205. Cham: Springer International Publishing.
- McCormick, J.; Hutchison, S.; Vincs, K.; and Vincent, J. B. 2015. Emergent behaviour: Learning from an artificially intelligent performing software agent. In *21st International Symposium on Electronic Art*.
- McInnes, L.; Healy, J.; and Melville, J. 2018. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv e-prints*.
- Naugle, L. 2000. Reflections on dancing with William Forsythe. <http://dance.arts.uci.edu/linaugle/files/mocap/forsythe/index.html>.
- Noland, C. 2009. *Agency and Embodiment: Performing Gestures/Producing Culture*. Harvard University Press.
- Pearson, K. 1901. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine* 559572.
- Schlkopf, B.; Smola, A.; and Miller, K.-R. 1998. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation* 10(5):1299–1319.
- van der Maaten, L., and Hinton, G. 2008. Visualizing high-dimensional data using t-SNE. *Journal of Machine Learning Research* 2579 – 2605.
- Warburton, E. C. 2011. Of meanings and movements: Re-languaging embodiment in dance phenomenology and cognition. *Dance Research Journal* 43(2):65–83.

Extending the Creative Systems Framework for the Analysis of Creative Agent Societies

Simo Linkola and Anna Kantosalo

Department of Computer Science

University of Helsinki

{simo.linkola, anna.kantosalo}@helsinki.fi

Abstract

Design and analysis of creative agent societies is often done in the context of computational sociology, which has evolved into its own field called computational social creativity (Saunders and Bown 2015). In this paper, we propose a formal framework for describing and analysing creative agent societies based on the Creative Systems Framework (CSF) (Wiggins 2006a; 2006b). We extend the CSF for single agents to include creative agent societies. The extended CSF allows us to describe society wide phenomena relevant for creativity, identify how individual agents relate to the whole society and characterise societal consequences caused by adopting certain policies. All these formal tools may be used when analysing designated creative agent societies. We demonstrate this by devising a straightforward practical procedure which may be used to gain insight into the influence individual agents have on the society over time.

Introduction

The Creative Systems Framework (CSF) (Wiggins 2006a; 2006b), defines exploratory (and transformational) creativity (Boden 1992) with mathematical rigour. It provides tools for describing and analysing diverse interesting phenomena, which may occur in creative systems. In this paper, we propose a minimal extension to the CSF, allowing us to utilise the CSF's formalisms to conduct extensive analysis on creative agent societies where each agent exhibits exploratory creativity. We call this extension the CSF for creative agent societies.

Computational social creativity (CSC) (Saunders and Bown 2015) is often adopted as a general context when analysing creative agent societies, i.e. multiple creative systems interacting with each other. In CSC, models should (1) demonstrate a mechanism, (2) be simple and reproducible, and (3) preferably generate new hypotheses (Saunders and Bown 2015). By a mechanism we mean that some properties of the individual systems or their interaction provoke observable emergent behaviour in the whole agent society, bringing into focus how various policies regarding artefact exchange and evaluation affect the society. Thus, demonstrating a mechanism implies that the preferred method used to analyse the society is of qualitative nature.

A prominent theoretical background used in CSC is Csikszentmihalyi's systems view of creativity (Csikszentmihalyi 1988), which has later been developed into the Domain-Individual-Field Interaction (DIFI) model. Its main argument is that creativity is not isolated in any individual. Instead, creativity can be understood in its entirety only by studying dynamic interactions between *individual* producers in the society, experts of a given *field*, and a *domain* of accumulated cultural artefacts (filtered by the field). This view can be seen as an encompassing conceptualisation of the creative process in a society, which itself allows elaborating on any of the following perspectives: what kind of artefacts are accepted into the domain, how a field is formed and how it operates, what are the abilities of the individuals, and how they create.

The Creative Systems Framework is a potent tool for describing and analysing individual creative systems exhibiting exploratory creativity. The CSF has been used as a foundation in several other studies (Grace and Maher 2015; Kantosalo and Toivonen 2016; Alvarado and Wiggins 2018). As a high level framework, the CSF does not take a stance on exactly *how* the exploratory creative process is executed. Rather, its formalisms allow describing and analysing the overall *capabilities* of a creative system's exploratory process, making it possible to identify different phenomena relating to these capabilities.

This paper expands the CSF's mathematical analysis tools into the context of CSC and the DIFI model. The extension to the CSF is designed for two purposes. First, it defines concepts which make it possible to discuss the capabilities of exploratory creative agent societies in detail. Second, it is a conceptual tool for analysing how changes in the properties of the individuals or the field affect the encompassing dynamics of the exploratory creative agent society. As such, it provides a novel point of view to CSC for inspecting if a particular mechanism exists or how the mechanism affects the whole society. The extension can be also utilised to gain insight into what kind of impact individuals or groups of individuals have on the field (e.g. indirect analysis on individuals' role in the society), or how the society's creative capabilities change over time (e.g. how the implemented interactions between the field and the individuals affect the behaviour of the whole society).

The rest of the paper is organised as follows. We begin

\mathcal{U}	the universe of all possible concepts
\mathcal{L}	a language in which to express concepts and rules
$\llbracket \cdot \rrbracket$	a function generator which maps a subset of \mathcal{L} to a function which associates elements of \mathcal{U} with a real number in $[0, 1]$
$\langle\langle \cdot, \cdot, \cdot \rangle\rangle$	a function generator, which maps three subsets of \mathcal{L} to a function that generates a new sequence of elements of \mathcal{U} from an existing one
\mathcal{R}	rules defining valid concepts
\mathcal{T}	rules defining traversal in the concept space
\mathcal{E}	rules defining evaluation of concepts

Table 1: Descriptions of different elements of the CSF.

by providing a brief introduction to the CSF, after which we move on to our contributions. First, we introduce our extension to the CSF, the CSF for creative agent societies, where the nucleus is an abstract societal aggregation function which outputs society wide representations of the CSF’s formalisms. Second, we show how the societal aggregation function and the society wide representations can be used to describe and identify various phenomena, both at a single time step and evolving in time, which may occur in creative agent societies. Lastly, we demonstrate that it is possible to derive practical procedures to test hypotheses considering the impact individual agents, or agent subsocieties, have on the whole society and its evolution. The paper ends with discussion and conclusions.

The CSF in a Nutshell

The Creative System’s Framework (CSF) (Wiggins 2006a; 2006b) is a formalisation of Boden’s (1992) exploratory and transformational creativity. To keep the scope of the paper reasonable, we will concentrate on exploratory creativity. However, we acknowledge that transformational capabilities of individuals are a major cause of emergent phenomena at the societal level, e.g. the field changes based on the changes in the individuals forming it or interacting with it.

According to Wiggins (2006a; 2006b), the exploratory part of the CSF is a septuple

$$\langle \mathcal{U}, \mathcal{L}, \llbracket \cdot \rrbracket, \langle\langle \cdot, \cdot, \cdot \rangle\rangle, \mathcal{R}, \mathcal{T}, \mathcal{E} \rangle.$$

The individual elements of the septuple are defined in Table 1. For full definitions, we refer to Wiggins (2006a; 2006b). The universe \mathcal{U} defines all possible (partial) concepts, i.e. artefacts, and a language \mathcal{L} provides means to express those (partial) concepts. In this paper we concentrate on \mathcal{R} , \mathcal{T} and \mathcal{E} , which are elaborated below.

\mathcal{R} is a set of rules (expressed in \mathcal{L}) for selecting acceptable artefacts. Applying a selector function generated from \mathcal{R} by the interpretation function $\llbracket \cdot \rrbracket$ (and a proper threshold $p \in [0, 1]$), gives us a way to define which artefacts are considered acceptable or valid by the agent (e.g., art, poetry, etc.). This formulation yields Boden’s (1992) conceptual space. Drawing from Kantosalo and Toivonen (2016), we denote the valid subset of concepts in the universe as

$$R \equiv \{c \mid c \in \mathcal{U} \wedge \llbracket \mathcal{R} \rrbracket(c) \geq p\}.$$

\mathcal{T} is a set of rules (expressed in \mathcal{L}), which realise, when interpreted, the traversal behaviour of the agent in the conceptual space. That is, it gives the means for the agent to move from known (even empty) concepts to unknown ones. The traversal may be informed by \mathcal{R} or \mathcal{E} , hence the interpretation function $\langle\langle \cdot, \cdot, \cdot \rangle\rangle$ is used to interpret the rules as behaviour. That is, given a (possibly empty) concept c_{in} , $\langle\langle \cdot, \cdot, \cdot \rangle\rangle$ outputs a new concept: $c_{out} = \langle\langle \mathcal{R}, \mathcal{T}, \mathcal{E} \rangle\rangle(c_{in})$. In general c_{in} and c_{out} can also be sets of concepts. We follow Kantosalo and Toivonen (2016) and denote the concepts reachable from a given concept c_{in} using at most n recursive steps as $T^n(c_{in})$:

$$T^n(c_{in}) \equiv \bigcup_{j=0}^n \langle\langle \mathcal{R}, \mathcal{T}, \mathcal{E} \rangle\rangle^j(c_{in}),$$

leaving out the superscript n when it is not required. Traversal behaviour defined in this manner is much alike the standard AI search framework.

\mathcal{E} is a set of rules (expressed in \mathcal{L}), which define the evaluation of the creative outputs, appropriately contextualised. This contextualisation might be subjective to the agent or some objective comparison. Similarly to conceptual space R , a set of *valued concepts* E may be defined by applying a selector function generated from \mathcal{E} by $\llbracket \cdot \rrbracket$ (and a proper threshold $p \in [0, 1]$):

$$E \equiv \{v \mid v \in \mathcal{U} \wedge \llbracket \mathcal{E} \rrbracket(v) \geq p\}.$$

To summarise, \mathcal{R} defines what kind of artefacts are accepted, \mathcal{E} defines which artefacts are valued, and \mathcal{T} provides the means to explore the artefact space. A prominent aspect which separates the CSF from standard AI search is the use of both \mathcal{R} and \mathcal{E} . For example, an independent creator may value slightly different artefacts than which it perceives as acceptable, creating ”tension” between \mathcal{R} and \mathcal{E} . With proper abilities of the agent, this tension may cause transformation in one of them, changing the exploration process in a fundamental manner. Especially transformation of \mathcal{R} , i.e. what types of artefacts are considered as acceptable solutions, is not present in standard AI search. This tension allows Wiggins to identify two types of interesting phenomena, namely *uninspiration* and *aberration* (Wiggins 2006a) driving transformations in the system.

Next, we will move on to describe our extension to the CSF, the CSF for creative agent societies, and how it may be used to describe interesting phenomena, similar to aberration and uninspiration, on a societal level.

The CSF for Creative Agent Societies

In this section we define our extension to the CSF. We begin by covering basic notation and assumptions. Then, we show how the CSF is interpreted for single agents with the addition of the input parameters presenting information exchange in the society, after which we move on to describe the societal elements of our extension: the societal aggregation function Π and the societal R , T and E .

The proposed extension is not ”complete” in the manner of the CSF, as we do not provide a full formalism for creative agent societies. Instead, we present the minimal elements

required to (1) discuss exploratory creativity on a societal level and (2) describe and analyse creative agent societies.

Our extension has two principal assumptions considering the agents in the creative society. First, each agent is considered as *an independent creator*. While the other agents may have an influence over the agent, ultimately the agent itself is in charge of its own creative process. Second, each agent in the society is assumed to exhibit exploratory creativity. This type of creativity is commonly used in creative agent societies, and it can be implemented with diverse generative methods. For example, genetic algorithms, many deep learning models and several other methods under the generate-and-test paradigm (Toivonen and Gross 2015) can be harnessed to obtain exploratory creativity. However, despite these design characteristics, many of the analysis tools presented in this paper can be applied with careful consideration to societies composed of other kinds of agents.

We define a society \mathbf{S} as a set of agents, $\mathbf{S} = \{A_1, A_2, \dots, A_n\}$, our standpoint being that inter-agent relationships, norms and other social phenomena are fundamentally (encapsulated in) the properties of the agents. Even though there might be some external representations of these concepts, the agents may (mis)interpret them. Thus, the state of the society and how it operates is functionally described by taking into account each agent's own view of the social structures.

We denote a time step the society \mathbf{S} is going through as t , $t \geq 0$, where the society (or a part of it) is initialised at $t = 0$. The semantics of the time (e.g. is the simulated time continuous or discrete) may vary between actual agent society implementations, however, the formulations in this paper are independent of them. Thus, the society \mathbf{S} on a time step t , \mathbf{S}^t , is a set of agents

$$\mathbf{S}^t = \{A_1^t, A_2^t, \dots, A_n^t\}.$$

We omit the superscript notation specifying the time step t when it is not necessary.

We assume that each agent $A_i \in \mathbf{S}$ is an independent creator with its own private CSF, meaning that the agent itself controls which artefacts it creates. Other agents may have an impact on the agent's creative process, but only the agent itself executes it. We denote the CSF of agent A_i on the time step t as

$$\langle \mathcal{U}, \mathcal{L}, \llbracket \cdot \rrbracket, \langle \langle \cdot, \cdot \rangle \rangle, \mathcal{R}_{A_i^t}, \mathcal{T}_{A_i^t}, \mathcal{E}_{A_i^t} \rangle,$$

assuming that the first four elements are the same for each agent, i.e. the agents use the same language to describe the concepts in the universe, and have the same interpreter functions. The agents only differ in what they consider as acceptable artefacts, the rules of their traversal behaviour and how they evaluate artefacts.

The agents need to be able to communicate with each other in a society. In creative agent societies, agents typically communicate by exchanging artefacts and the artefact producer's identity may affect the perception of the artefact. Thus, we need to alter the interpreter functions $\llbracket \cdot \rrbracket$ and $\langle \langle \cdot, \cdot \rangle \rangle$ (as agents may also start their search from (partial) artefacts given to them by their peers) to handle this requirement. We extend both $\llbracket \cdot \rrbracket$ and $\langle \langle \cdot, \cdot \rangle \rangle$ to accept a second input argument, the identity (or a set of identities) of the agent

which produced the artefact, I . As a consequence, agent's A_i conceptual space R_{A_i} is defined as

$$R_{A_i} \equiv \{c \mid c \in \mathcal{U} \wedge \llbracket \mathcal{R}_{A_i} \rrbracket(c, I) \geq p, \text{ where } I \subseteq \mathbf{S}\},$$

E_{A_i} is handled in the same way, and $T_{A_i}^n$ is defined as

$$T_{A_i}^n(c_{\text{in}}, I) \equiv \bigcup_{j=0}^n \langle \langle \mathcal{R}_{A_i}, \mathcal{T}_{A_i}, \mathcal{E}_{A_i} \rangle \rangle^j(c_{\text{in}}, I).$$

We assume that all other communication between the agents (other metadata related to the artefacts, such as framing, communication about beliefs and intentions, etc.) is handled through some other properties or processes of the agents, which are not explicitly presented in the extension. This communication may, however, drive the change in the agent's own creative process, thus transforming some elements of its CSF, e.g. E_A or T_A .

We denote by $R_{A_i^t}$ the set of artefacts considered as valid by A_i on time step t , and by $E_{A_i^t}$ the set of artefacts considered valuable. For creativity, it is especially interesting, when these sets are subject to alterations over time. These alterations can be due to the change in the set of rules an agent uses or the threshold p applied to the outputs of the function interpreted from the rules, e.g. $\llbracket \mathcal{R}_{A_i} \rrbracket$.

II: The Societal Aggregation Function

In the centre of the proposed extension is a societal aggregation function Π , which interprets the properties of the individual agents (their CSFs, relationships, etc.) into society wide concepts, taking into account the different social norms, processes and policies present in the society. By abstracting the interaction between the agents, their relationships, etc., into Π , we lose a lot of meaningful information about a particular society, but gain the elegance of the CSF in describing interesting situations which may arise in it.

Formally, the societal aggregation function Π takes as an argument a set of agents at time step t , \mathbf{S}^t , and outputs societal R , T and E for that time step. That is,

$$\Pi(\mathbf{S}^t) = \{R_{\mathbf{S}^t}, T_{\mathbf{S}^t}, E_{\mathbf{S}^t}\},$$

where the societal structures $R_{\mathbf{S}^t}$, $T_{\mathbf{S}^t}$ and $E_{\mathbf{S}^t}$ are an aggregation of all the individual agents' properties at time step t affected by \mathbf{S} 's policies and other societal structures. In the special case when $\mathbf{S} = \{A\}$, i.e. the society contains only a single agent, Π returns the agent's own R_A , T_A and E_A . The societal aggregation function Π acts as an interpreter function for the individual agents and their interactions. Its outputs, $R_{\mathbf{S}}$, $T_{\mathbf{S}}$ and $E_{\mathbf{S}}$, define the (implicit or explicit) social interpretations of which artefacts society perceives as valid, what artefacts can be reached by the society, and which artefacts are valued by the society, respectively.¹

¹The fact that Π returns the interpreted $R_{\mathbf{S}}$, $T_{\mathbf{S}}$ and $E_{\mathbf{S}}$ and not the societal rules $\mathcal{R}_{\mathbf{S}}$, $\mathcal{T}_{\mathbf{S}}$ and $\mathcal{E}_{\mathbf{S}}$ still requiring interpretation is purely a design decision. One could also formulate Π to return the latter and define the societal interpretation function for each set of rules. Most of the analysis tools described later in the paper could be very well applied to the sets of rules instead of the sets of artefacts, providing yet another angle to the creative agent society's operation. However, hierarchical societies could benefit from first composing the societal rules and then their interpretations.

Next, we characterise R_S , T_S and E_S and how they relate to the DIFI model's concepts. In the following, we will assume that the society has only one field, and all agents may be part of it. That is, the agents in the society do not form isolated subsocieties and they produce concepts within the same domain, e.g. art.

R_S : The Societal R for S

R_S is the society S 's collective understanding of which artefacts are accepted as valid, e.g. works of art. Transforming the individual agent properties into R_S has to take into account how the society's policies, communication structures, etc., aggregate individual agents' views of what is accepted as a societal norm of art. However, as our goal is to be able to describe the capabilities of the creative agent society, Π is designed to abstract away exactly the decisions related to how R_S should be formed.

The aggregation of individual properties into societal properties may be implicit or explicit. In the implicit case there is no clearly defined society wide decision procedure for acceptance, whereas in the explicit case there is a procedure (to which agents more or less conform to) which outputs acceptance for each artefact or a fuzzy assignment for it. In the explicit case, R_S is the output of the procedure if it would be run for all artefacts. However, as this is not computationally feasible, typically only the artefacts perceived by the society on a given time step are put into test for inclusion. In the implicit case, the analyst should define a proper way to compute R_S from the properties of the individual agents. From an agent's perspective, the characteristic difference between the two is that the output of the explicit aggregation (given the input) is known to the agent, while in the implicit case the agent needs to model the process by some means.

In practice, the explicit decision procedure might be, e.g., some aggregate measure taking into account each agent's individual R_A and voting for inclusion in R_S for each candidate artefact. Another way would be to compute each agents $\llbracket R_A \rrbracket$ for each artefact, producer identity pair and applying some aggregate function (such as min, max or mean) to it, which is then thresholded to filter artefacts accepted to R_S .

From the DIFI model's point of view, R_S can be seen to be formed by the field. On the other hand, a set of agents with reasonably similar R , which differs from R_S , may form their own field, specialising to a particular type of artefacts. The experts in the field define which artefacts are seen as art and, thus, are candidates for domain inclusion. However, this process is dynamic. First, the domain affects the perception of new artefacts, each new artefact included in the domain has an impact on the perception of subsequent artefacts. Second, the identity of individuals producing artefacts affect their social evaluation. Third, the individuals which form the field may change, thus affecting the process.

T_S : The Societal T for S

T_S encapsulates the artefacts reachable by the society. Each agent has its own \mathcal{T}_A and T_A , but how all agents traverse the space is affected by the communication between the agents. An agent A_i communicating an artefact it has (partially) produced (or is aiming to produce) to another agent A_j may

cause alterations to R_{A_j} , T_{A_j} or E_{A_j} , which in turn may result in A_j communicating back to A_i . This kind of cyclic influence can drive the exploration constantly to new areas, especially if the agents are deliberately seeking to produce artefacts seen as novel by their peers.

In a social setting such artefact exchanges between individuals may also directly affect what is created next. For example, an agent A may take an artefact produced by another agent as its own creation process' starting point (see Kantosalo and Toivonen (2016)), effectively moving to a different area in the conceptual space. This area could even be unreachable for an agent drawing inspiration only from its own productions during its creative process.

As a society's exploration of the artefact space is a conjoined process of the individual exploration processes of its agents, it is not explicitly contained in any single component of the DIFI model. Instead, it is a dynamic process where individuals aim to produce creative artefacts, but at the same time are interacting with the field which evaluates them. Further, the individuals gain information from the domain which may alter their own creative process and goals. All these properties can affect which artefacts the society can reach, i.e. T_S .

E_S : The Societal E for S

E_S defines which artefacts have society wide value. In some societies, E_S may have a similar relation to single agent evaluation E_A , as historical creativity (H-creativity) has to psychological creativity (P-creativity) (Boden 1992). In these societies, the artefacts evaluated highly by E_S are typically more likely to exhibit H-creativity, whereas single agent's E_A reflects merely the agent's own personal view of the artefact. However, this might not always be the case. For example, the evaluations given by a host of commoners may favour familiarity more than the evaluations given by a few experts, making it possible for E_S to become biased towards more mundane artefacts, like pop songs.

As E_S defines the artefacts which are valued by the society, it has an obvious connection to the domain in the DIFI model. The domain is a collection of cultural artefacts which have been perceived as valuable at some point during the society's lifetime. These artefacts are filtered by the field, i.e. they are subject to the social decision making policies existing in the society, and thus are also affected by the properties of the individuals.

Analysing Creative Agent Societies

The proposed extension can be utilised in various ways to describe and analyse relevant situations which may occur in creative agent societies. We begin by defining the term *substantial change* (to R_S , T_S or E_S), which is utilised throughout this section. Then, we provide descriptions of the society on a single time step and how the society changes over time. We continue by showing how different agent roles, or the impacts agents have on the society, can be identified through our extension. Lastly, we sketch out some ways in which the extension can be used to analyse the policies present in the society.

We propound that the abstract analysis tools described below can be put to practice by the society's designer or analyst by defining proper (qualitative or quantitative) measures for R_S , T_S , E_S and substantial change to each of them. This will highlight what kind of effects particular mechanisms, policies or agent properties have on the society. We demonstrate this by presenting a practical procedure for agent role analysis in the next section. Although the formulations in this section have been done using set notation, we envision that in practice similar ideas can be formulated also using, e.g. probability calculus.

Substantial change For analysis purposes, we define the term *substantial change* (to R_S , T_S or E_S), to describe an alteration which has a considerable effect on the society's operation. In CSC, this impact would preferably be of qualitative nature. For example, a substantial change to R_S could mean that the society's perception of what is considered as art would be significantly adjusted, e.g. a new painting style would be accepted as art. For R_S and E_S , a necessary but not sufficient criteria for a substantial change is that some artefact which was not included before the change is included in the set after the change or vice versa. The nature of T_S is slightly different, as it considers artefacts reachable by the society. We envision that for T_S the analyst may also want to take into consideration how likely it is for the society to reach specific artefacts, as some artefacts may be reachable by several individuals.

Single Time Step Analysis

In this section, we provide some general descriptions of situations which consider the whole creative agent society S on a single time step t . First, we describe two situations drawn from Kantosalu and Toivonen (2016) and then two situations characterising the relation between R_S and E_S .

Societal conceptual mismatch and artistic disagreement

A *societal conceptual mismatch* occurs when the society S in its entirety cannot agree on which artefacts it considers as valid, resulting in an empty conceptual space: $R_S = \emptyset$. Similarly, a *societal artistic disagreement* occurs when the society S cannot agree on which artefacts it considers as valuable, resulting in the empty set of valued artefacts $E_S = \emptyset$.

For both of the situations, an interesting case is when S is composed of two separate subsocieties, which both have a non-empty internal R (or E), but are not able to form a collective understanding of it. Formally, for such societal conceptual mismatch, there exist two subsocieties $G, H \subset S$, where $G \cup H = S$ and $G \cap H = \emptyset$, such that $R_G \neq \emptyset$, $R_H \neq \emptyset$ and $R_{G \cup H} = \emptyset$.

Both societal conceptual mismatch and societal artistic disagreement may originate from multiple sources. First, the agents' individual views of R or E may be simply too different, e.g. agent A_i may not value anything even remotely close to what is valued by agent A_j , making it challenging for A_i and A_j to reconcile their views. Second, the policies present in the society may not account for minuscule changes in the agents' views, e.g. when the societal

policy requires unanimity for inclusion, but every artefact is rejected by exactly one agent.

However, these situations are not always detrimental for the creative agent society. For example, in a society where the agents cultivate their expertise by specialising in particular art styles, a societal conceptual mismatch may be caused by the field being divided into specialised subfields.

Harmonious and charged societies The relation between E_S and R_S is interesting for the creative agent society as the tension between these two concepts may be a prominent reason for the society's transformation. We characterise two idealised situations which may occur in a society S .

A society S is harmonious if $E_S = R_S$. This means that the society values exactly the artefacts it perceives as valid. This may look like a favourable situation for the society, but if the situation prolongs, it may cause stagnation as the society has reached an equilibrium where neither E_S nor R_S provokes changes in the other.

In a charged society $R_S \neq E_S$ and there needs to be substantial change to either E_S or R_S to make them equivalent. That is, there is some social "tension" between E_S and R_S , which may provoke cultural intercourse demanding changes to either E_S or R_S . This provides a natural cause for transformation in the society.

Dynamic Analysis

The temporal transformations of creative agent societies are one of the key interests in computational social creativity (Saunders and Bown 2015). These emergent, society wide phenomena can only be observed when the agent society is executed for a reasonable time span. In this section we present some fundamental characterisations for creative agent societies requiring the time dimension.

Stagnant society A society S has a stagnant X_S , where $X_S \in \{E_S, R_S, T_S\}$, from time step t onward, if and only if for all $k > 0$ there is no substantial change from X_{S^t} to $X_{S^{t+k}}$. That is, a stagnant society is not able to produce a substantial change to R_S , T_S or E_S in any period of time.

For example, a stagnant R_S could mean that either the agents themselves are not able to change their own R_A , or the society's social processes cannot account for changes in the individual agent's R_A . In both cases the society wide outcome is the same, although the means to escape it differ.

A society which is stagnant in one or two of its societal elements, E_S , R_S and T_S , may still be able to transform in a meaningful manner. The society reaches a pathological state only when all three of these societal elements stagnate. The society ceases to evolve, limiting its ability to produce creative artefacts in the future.

Continuously transforming society A society S has a continuously transforming X_S , where $X_S \in \{E_S, R_S, T_S\}$, if and only if for all $t \geq 0$ there exists $k > 0$ for which $X_{S^{t+k}}$ is substantial change from all X_{S^a} , where $0 \leq a \leq t$. That is, a continuously transforming society is always able to produce an alteration (to R_S , T_S or E_S) which is a substantial change from all the situations (w.r.t R_S , T_S or E_S) the society has previously gone through.

Diverging society The society S has a diverging X_S , where $X_S \in \{E_S, R_S, T_S\}$, from time step t to $t+k$, if there exists two subsocieties $G, H \subset S$, where $G \cap H = \emptyset$, such that $X_{G^{a+1}} \cap X_{H^{a+1}} \subseteq X_{G^a} \cap X_{H^a}$ for all $t \leq a < t+k$ and $X_{G^{t+k}} \cap X_{H^{t+k}}$ is a substantial change to $X_{G^t} \cap X_{H^t}$ and $X_{H^{t+k}}$ is a substantial change to $X_{G^{t+k}}$. That is, in a diverging society the two subsocieties G and H monotonously differ on their X_G and X_H , ultimately producing a substantial change on what artefacts they include (in case of R and E) or can reach (in case of T). A diverging society may end up in a societal conceptual mismatch or an artistic disagreement.

Charging society The society S is charging from time step t to $t+k$, if $E_{S^{a+1}} \cap R_{S^{a+1}} \subseteq E_{S^a} \cap R_{S^a}$ for all $t \leq a < t+k$ holds and $E_{S^{t+k}} \cap R_{S^{t+k}}$ is a substantial change to $E_{S^t} \cap R_{S^t}$ and $E_{S^{t+k}}$ is a substantial change to $R_{S^{t+k}}$. That is, in a charging society R_S and E_S are diverging from each other, ultimately producing a substantial change to what artefacts they include, i.e. the society becomes charged.

Analysing Agent Roles and Impact on the Society

Analysing agent roles or their impact on the society is focal to CSC. We present two ways to describe and analyse individual agents, their relation to the whole society and their influence on it: *relation method* and *alteration method*. Both of the methods can be used either in a single time step or for a dynamic analysis of the agent society. We restrict our analysis in this section to individual agents, but it is straightforward to generalise these methods to also consider agent subsocieties.

Relation method The relation method compares the properties of a single agent A to the society wide properties, e.g. how E_A differs from E_S . This method can be used to describe how the agent's properties relate to the whole society, but fails to capture the actual influence the agent has on the society via interactions and other social processes.

We define three agent types based on the single time step relation between the agent's $X_A \in \{R_A, E_A\}$ and the whole society's counterpart $X_S \in \{R_S, E_S\}$: *contained*, *controversial* and *contradicting*.

A contained agent A has its own understanding of X_A fully incorporated in the society's interpretation. Formally, $X_A \subset X_S$. That is, the society agrees on the agent's view, but the agent might not agree on other artefacts belonging to X_S (especially if X_S is a substantial change to X_A).

A controversial agent A has only a part of its own understanding of X_A incorporated in the society's interpretation. Formally, $X_A \cap X_S$ is substantial change to X_A , X_S and \emptyset . This means, that there is something meaningful that is left out from the intersection from both the agent's and the society's perspective, and the intersection itself contains some meaningful set of artefacts.

A contradicting agent A has none of its own understanding of X_A incorporated in the society's interpretation. Formally, $X_A \cap X_S = \emptyset$. In most of the cases this also implies that the agent has a conceptual mismatch (if $X = R$) or artistic disagreement (if $X = E$) with the whole society S .

With the dynamic relation method we are able to specify if an agent A is merging or diverging from a society with respect to $X_S \in \{R_S, E_S\}$. A diverging agent is actively moving away from X_S while a merging agent is actively moving inward into X_S . Formally, an agent A is merging to a society S with respect to X_S from time step t to time step $t+k$, if it holds that $X_{A^a} \cap X_{S^a} \subseteq X_{A^{a+1}} \cap X_{S^{a+1}}$ for all $t \leq a < t+k$ and $X_{A^{t+k}} \cap X_{S^{t+k}}$ is a substantial change to $X_{A^t} \cap X_{S^t}$, and $X_{A^{t+k}}$ is a substantial change to X_{A^t} , while $X_{S^{t+k}}$ is not a substantial change to X_{S^t} . That is, X_A is actively becoming more similar to X_S while X_S does not change in a relevant manner. Ultimately, merging may cause a contradicting agent to become contained.

Alteration method The alteration method can be used to analyse what is the influence of (a property of) an agent on the whole society. In the single time step alteration method, an agent A is altered to agent A' and the society wide interpretations between the unaltered society S and the altered society U are compared. In the dynamic alteration method, the alteration from A to A' is thought to occur on a time step t , after which emergence in the unaltered society S and the altered society U are compared.

A particularly strict version of alteration method is acquired when the agent A is removed entirely from the society. We call this version of alteration method the *subtraction method*.

Sosa and Gero (2005) identify *gatekeeper* agents. These agents act as opinion leaders which have a high impact on the artefacts filtered into the domain. We formulate a gatekeeper as an agent A for which the subtraction method (or appropriately devised alteration method) produces a substantial change to E_S . Formally, an agent A is a gatekeeper if $E_{S \setminus \{A\}}$ is a substantial change to E_S .

Sosa and Gero (2005) discuss *change agents* which are the agents driving the social change in the society. We formulate a change agent as an agent which is likely to cause a substantial change to R_S , T_S or E_S , by exploiting the dynamic alteration method. Formally, an agent A is a change agent for E_S if there is substantial change from E_{S^t} to $E_{S^{t+k}}$, but by modifying A to A' from time step t onward, thus altering the society S to society U on time step $t+1$, there is no substantial change from E_{S^t} to $E_{U^{t+k}}$. That is, without the change agent the substantial change does not occur. (It would be beneficial to ensure that the substantial change does not occur before time step $t+k+m$ for some reasonable m , but we leave it out for clarity.)

Analysing Societal Aggregation

Our extension allows indirect observation of the effects different social policies (integrated in the societal aggregation function Π) have on the society. We sketch out a few ways in which we envision the extension to be exploited for social policy analysis.

The most straightforward way is to compare the effects two policies have on the society's R_S , T_S and/or E_S . That is, given society S , we change some of its policies to obtain society S' and then compare $\{R_S, T_S, E_S\}$ to $\{R_{S'}, T_{S'}, E_{S'}\}$. This allows evaluating, e.g. if it is possible to get rid of a

stagnant R_S by changing the policies within the society in a particular way.

Another way is to compare how each agent affects the whole society and to draw some insight into the policy from the comparison. For example, we can exploit a single time step subtraction method for this: for all $A_i \in \mathbf{S}$ compare $E_{\mathbf{S} \setminus \{A_i\}}$ and E_S . If none of $E_{\mathbf{S} \setminus \{A_i\}}$ are a substantial change to E_S , then there are no gatekeepers in the society. This could suggest that the society is somewhat balanced in its social decision making processes considering E_S .

Lastly, we can compare how each agent's R , T or E relate to other agents' counterparts or to the society wide R_S , T_S or E_S . For example, the socially novel goal selection (Hantula and Linkola 2018) could be characterised so that for each agent $A \in \mathbf{S}$ it is likely that for each of its peers $B \in \mathbf{S}$, where $B \neq A$, holds that E_B is a substantial change to E_A . That is, it is likely that each agent has an evaluation method which differs substantially from all other agents, or at least the agent pairs for which this does not hold are scarce.

Practical Procedure for the Alteration Method

To show that the analysis tools introduced above can be put into practice, we present an example procedure for the alteration method. The procedure allows the analyst of a creative agent society to test various hypotheses related to the agents and their effect on a specific society \mathbf{S} from time step t onward, i.e. on $E_{\mathbf{S}^{t+k}}$, $R_{\mathbf{S}^{t+k}}$ and/or $T_{\mathbf{S}^{t+k}}$.

Particularly, the procedure allows the analyst to collect evidence if suspected agents (or agent properties) are indeed causing the observed emergence in the society or would the changes occur even without the agents (or agent properties) assumed to cause it. As a consequence, new hypotheses for what causes the emergence or how the dynamics of the society function may be formed.

The procedure has two main assumptions. First, it assumes that the agent society is "closed", i.e. the analyst is able to control all the parameters (and thus all the agents) within the simulation. Second, all the agents (subject to distinct hypotheses) should be alterable (even removable in the case of the subtraction method) during the simulation without breaking it.

The procedure contains the following steps:

1. Fix all random number generator seeds and other parameters of the simulation of the creative agent society \mathbf{S} .
2. Execute the simulation (simulation P_1).
3. Produce a hypothesis of the impact (some property of) an agent $A \in \mathbf{S}$ has on the society from a particular time step t onward.
4. Test the hypothesis by running the simulation on \mathbf{S} with the same parameters again and altering A on time step t (simulation P_2).
5. Analyse how the two simulations, P_1 and P_2 , differ from each other from time step t onward.
6. Repeat the steps 1-5 several times with different random number seeds to obtain statistical results.

The procedure requires the parameters to be fixed so that the exact same simulation can be run again. In this way, the only thing which changes between the two simulations is the

alteration that is done to the agent. With a proper measure of (some of) the society's interpretations (R_S , T_S and/or E_S) it is possible to analyse if the simulations P_1 and P_2 differ substantially from time step t onward, suggesting that the alteration made to agent A was its cause. As CSC models should generally be simple (Saunders and Bown 2015), it should be reasonable to verify if this was indeed the case.

In practice, it may be challenging to devise alterations which do not change the state of the random number generators and still negate the effects of the agent properties which are under consideration. In this situation, the simulation P_2 should be run a number of times to be able to clearly conclude that the alteration produces different kind of emergence than what is observed in P_1 .

Discussion

Overall, one can distinguish four different perspectives to creativity, each of which may be adopted when describing, analysing and assessing a system's creativity. The perspectives are: the creative Product (What properties make the output creative?), Process (How are creative results produced?), Producer (What kind of characteristics of systems lead to creative behaviour?), and Press (Who judges whether the product is creative, and how?) (Jordanous 2016).

Frameworks, definitions and other tools utilised in (computational) creativity typically emphasise one or two of the above mentioned perspectives. For example, the standard definition of creativity (Runco and Jaeger 2012), which defines creativity as an ability to produce outputs which are novel and valuable, highlights the Product (output) and the Producer (abilities) perspectives. On the other hand, defining computational creativity as "the art, science, philosophy and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviours that unbiased observers would deem to be creative" (Colton and Wiggins 2012) puts stress on the Producer (the system taking responsibilities) and the Press (unbiased observer is the final evaluator) perspectives. Moreover, Boden's three types of creativity (Boden 1992) adopt the Process perspective by considering how the artefacts are produced.

The CSF conforms to the illustrated standard. It formalises exploratory and transformational creativity (Boden 1992), and as such it can be seen to adopt the Process perspective, taking into account also the Product perspective on a conceptual level (the validation and evaluation of the artefacts using \mathcal{R} and \mathcal{E}). However, when it is utilised as a "stateless" conceptual framework, it is usually perceived to describe abilities of a creative system, i.e. to adopt the Producer perspective.

Following the style of the original CSF, the extension proposed in this paper is geared towards describing and analysing the overall capabilities of, or situations occurring in, a creative agent society, not the instantiated interactions taking place in the society or other transitory social phenomena. As such, the extension may be seen as counter-intuitive to the CSC point of view, where observing actualised interactions between the agents during the society's execution and identifying agent attributes causing particular emergent phenomena are of key importance (Saunders and

Bown 2015). However, the society wide capabilities can be perceived as abilities of the society which *emerge* from the single agent properties (the Producer perspective) and the social policies present in the society (the Press perspective).

We propound that the extension provides a novel point of view on creative agent societies, facilitating a mathematically formal analysis of the society's creative potential and how it changes over time. Even though the extension does not offer tools to analyse the exact communications taking place in an agent society, it provides the means to analyse the consequences of communication and other social phenomena on a conceptual level. Thus, it can be used in CSC to gain evidence to show if certain mechanisms exist, which is one of the main goals of CSC models (Saunders and Bown 2015), or to test hypotheses on the impact of particular agents (or their properties) on the society. Further, by providing an alternative perspective, the extension may provide insight into new hypotheses considering the creative agent societies that would elude the analyst otherwise.

Identifying interaction emergence which does not directly affect R_S , T_S or E_S , such as communication patterns arising in a society of curious agents (Saunders and Gero 2002), is challenging with the extension by design. Moreover, the proposed extension is just one of the many reasonable alternatives to formulate the CSF for creative agent societies. For example, it is possible to design creative agent societies where the agents would be merely generative, but the society in its entirety would appear creative. In this paper, we have formulated each agent to contain its own CSF to show the full analysis power of the extension, but it would be possible to remove some of the CSF's elements from the agents. For example, it is common in CSC models that the agents are given by design a fixed R which does not change over time.

Conclusions

We have proposed a minimal extension to the Creative Systems Framework (Wiggins 2006a; 2006b), the CSF for creative agent societies, where each agent is an independent producer exhibiting exploratory creativity.

We have shown the extension's strength as a conceptual tool by utilising it to (1) define phenomena, both instantaneous and temporal, relevant for creativity which considers the whole agent society, (2) describe individual agent's relation to and influence over the society, and (3) characterise the effects different social policies may have on the society. All the preceding aspects can be exploited in the analysis of creative agent societies. We have demonstrated that although these analysis tools derived from the extension are conceptual, and idealised in their nature, one can devise appropriate practical procedures for them.

In the future, we hope to enhance the extension's expressive power by including more agent oriented machinery into it. For example, it would be interesting to study how each agent's memory could be presented in the extension and how it would affect the society's transformative capabilities.

Acknowledgements. This work has been supported by the Academy of Finland under grant 313973 (CACS).

References

- Alvarado, J., and Wiggins, G. A. 2018. Exploring the engagement and reflection model with the creative systems framework. In *Proceedings of the Ninth International Conference on Computational Creativity (ICCC 2018)*, 200–207. Salamanca, Spain: Association of Computational Creativity.
- Boden, M. 1992. *The Creative Mind*. London: Abacus.
- Colton, S., and Wiggins, G. A. 2012. Computational creativity: The final frontier? In *Proceedings of the 20th European Conference on Artificial Intelligence, ECAI'12*, 21–26. Amsterdam, The Netherlands, The Netherlands: IOS Press.
- Csikszentmihalyi, M. 1988. Society, culture, and person: A systems view of creativity. In Sternberg, R. J., ed., *The Nature of Creativity: Contemporary Psychological Perspectives*. Cambridge University Press. 325–339.
- Grace, K., and Maher, M. L. 2015. Specific curiosity as a cause and consequence of transformational creativity. In *Proceedings of the Sixth International Conference on Computational Creativity*, 260–267. Park City, Utah: Brigham Young University.
- Hantula, O., and Linkola, S. 2018. Towards goal-aware collaboration in artistic agent societies. In *Proceedings of the Ninth International Conference on Computational Creativity (ICCC 2018)*, 136–143. Salamanca, Spain: Association of Computational Creativity.
- Jordanous, A. 2016. Four PPPPerspectives on computational creativity in theory and in practice. *Connection Science* 28(2):194–216.
- Kantosalu, A., and Toivonen, H. 2016. Modes for creative human-computer collaboration: Alternating and task-divided co-creativity. In *Proceedings of the Seventh International Conference on Computational Creativity*, 77–84. Paris, France: Sony CSL.
- Runco, M. A., and Jaeger, G. J. 2012. The standard definition of creativity. *Creativity Research Journal* 24(1):92–96.
- Saunders, R., and Bown, O. 2015. Computational social creativity. *Artificial Life* 21(3):366–378.
- Saunders, R., and Gero, J. S. 2002. How to study artificial creativity. In *Proceedings of the Fourth Conference on Creativity & Cognition*, 80–87. Loughborough, UK: ACM.
- Sosa, R., and Gero, J. S. 2005. Social models of creativity. In *Proceedings of the International Conference of Computational and Cognitive Models of Creative Design VI*, 19–44. Heron Island, Australia: Key Centre of Design Computing and Cognition, University of Sydney, Australia.
- Toivonen, H., and Gross, O. 2015. Data mining and machine learning in computational creativity. *Wiley Int. Rev. Data Min. and Knowl. Disc.* 5(6):265–275.
- Wiggins, G. A. 2006a. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems* 19(7):449–458.
- Wiggins, G. A. 2006b. Searching for computational creativity. *New Generation Computing* 24(3):209–222.

TwitSong 3.0: Towards Semantic Revisions in Computational Poetry

Carolyn Lamb and Daniel G. Brown

David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
c2lamb@uwaterloo.ca; dan.brown@uwaterloo.ca

Abstract

We present TwitSong 3.0, a found poetry system which locates candidate lines in a source text or generates them, then edits the lines repeatedly to increase their score on measures of meter, topicality, imagery, and emotion. We evaluate TwitSong 3.0 using a survey of domain experts. The system's editing process does significantly improve its lines, although the resulting poems are not always coherent, and the experimental evidence suggests that the improvement may not occur through the precise mechanisms we intended.

Introduction

This paper continues the work of poetry generation begun in our previous two papers (Lamb, Brown, and Clarke 2015; Lamb, Brown, and Clarke 2017), in which we use our TwitSong system to generate found poetry based on the news. TwitSong works by gathering candidate lines from a large topical corpus, rating them on various scales for their poetic suitability, and assembling them into a formal, rhymed poem. Prior versions of TwitSong produced mixed results; our motivation in the current study was to improve the system's foundations by making its underlying process more sophisticated.

The current generation of TwitSong introduces a new mechanism. TwitSong 3.0 is able to make targeted, goal-directed edits to its own work using our Editorial Algorithm, a form of genetic programming inspired by the human creative process. This builds on the work of Gervás (2016), which also edits candidate lines for traits such as rhyme and number of syllables. TwitSong 3.0 goes further by editing for semantic traits such as topicality and emotion.

We evaluate TwitSong 3.0 and find that using the Editorial Algorithm significantly improves the resulting poems in terms of expert judges' pairwise preferences. However, this improvement is small and much of it is due to the Editorial Algorithm's effect on meter. Alternatively, the improvement can be interpreted as a result of contradictory effects in our line selection criteria. While the overall quality of the poems is not what we hoped, TwitSong 3.0's evaluation also serves as an example of good practice for computational poetry evaluation, and as an application of the evaluation principles we have developed in our prior work (Lamb, Brown, and Clarke 2016; Lamb, Brown, and Clarke 2018).

Related Work

TwitSong (Lamb, Brown, and Clarke 2015; Lamb, Brown, and Clarke 2017) is a found poetry system in which human-written candidate lines are modified and recombined. Similar found poetry systems in the computational creativity literature include The Poet's Little Helper (Astigarraga et al. 2017) and DopeLearning (Malmi et al. 2015). Generating poetry based on the news is a poetic goal previously worked for by systems including P.O.Eticus (Toivanen, Gross, and Toivonen 2014) and Pemuisi (Rashel and Manurung 2014). Evaluation of earlier versions of TwitSong showed that rating lines on criteria such as topicality and emotion could produce overall better poems than a control (Lamb, Brown, and Clarke 2015), but that automating these ratings did not always produce the desired effect (Lamb, Brown, and Clarke 2017).

The idea of creativity as a generation-evaluation loop, with a creator able to repeatedly evaluate its unfinished work and make targeted improvements, is important in many theories of computational creativity and creativity psychology (Ward, Smith, and Finke 1999; García et al. 2006; Simonton 2011; Dahlstedt 2012). The use of looping, targeted edits for poetry specifically was previously done by Diaz-Agudo *et al.* (Díaz-Agudo, Gervás, and González-Calero 2002) with the COLIBRI system, and continued by Gervás with WASP (Gervás 2013a; Gervás 2013b; Gervás 2016). Various versions of these systems edit candidate lines for rhyme, meter, stress pattern, excessive similarity to the source text, excessive repetition, sentence length, verse length, and grammatical plausibility of the final word in the sentence. Gervás expresses a desire to use similar techniques to optimize for semantic traits, such as topicality (Gervás 2016), but no such method has yet been found. Apart from being included for publication in a book about computational poetry (Gervás 2013a), neither COLIBRI nor WASP have been formally evaluated.

Formal evaluation for computational creativity systems is a topic worthy of books in itself; our previous survey (Lamb, Brown, and Clarke 2018) gives a detailed interdisciplinary overview. Many computational poetry systems are not formally evaluated, or are evaluated without adhering to what we argue are best practices for evaluation. In this paper we focus on a few such best practices: the testing of falsifiable hypotheses about a system's creative output; the use

of domain expert judges; and the use of domain-specific, evidence-based testing criteria.

Our own prior research (Lamb, Brown, and Clarke 2016), analyzing the written responses of poetry quasi-experts to examples of computational poems, identified four major criteria that such experts look for when expressing their opinions about poetry. These criteria are Reaction (the expert's personal, emotional response to the poem), Meaning (the sense that the poem conveys a message), Novelty (the sense that the poem says something different from what has been said before), and Craft (the skill and technique with which the poem is constructed, including specific poetic devices). Each of these criteria is also divided into subcriteria. To our knowledge these are the first poetry evaluation criteria that have been developed directly from the study of poetry experts' responses, rather than stated ad hoc by the researcher or lifted from another creative domain. More work remains to be done on the four poetry criteria before they constitute a reliable and valid set of constructs for testing, but the same can be said of any other existing group of criteria, so for now we refer to the four criteria throughout our own research.

How TwitSong 3.0 works

For line representation, RhymeSet construction, line judging, and poem construction, TwitSong 3.0 is built on similar code to its previous generations (Lamb, Brown, and Clarke 2015; Lamb, Brown, and Clarke 2017). In brief, a source text is mined for potentially rhyming phrases of the appropriate length, and these phrases are grouped based on end rhyme. Syllabification and rhyme detection is performed using the CMU Pronouncing Dictionary and with Hirjee and Brown's Rhyme Analyzer code (Hirjee and Brown 2010). These representation, judgment, and construction mechanisms are not new; what is new in this generation is the Editorial Algorithm and its Markov chain-based targeted reconstruction of lines.

Each candidate line in a RhymeSet is given an automated score based on our line judgment criteria. For TwitSong 3.0, these are:

- **Emotion.** A target emotion is chosen for the poem based on prevalence in the source text and appropriateness for the topic. The line is then scored by adding together the scores of each of its individual words for this emotion in the NRC Hashtag Emotion Lexicon (Mohammad and Kiritchenko 2015), which was specifically developed for use with short texts like tweets or the lines of a poem. The goal with measuring Emotion is to produce an appropriate emotional reaction in the reader, for the Reaction criterion.
- **Imagery.** Each line is given a score for the concreteness of its imagery by adding together the scores of its individual words in the Regressive Imagery Dictionary (Provalis 1990), a dictionary used by Simonton (Simonton 1990) when statistically comparing more and less successful poems. "Primary process" words in the dictionary are associated with more concrete imagery and more successful poetry overall. Kao and Jurafsky (Kao and Jurafsky 2012), analyzing professional and amateur contemporary

poetry with a similar tool, also found that concrete imagery was one of the strongest predictors of more successful professional poetry. Imagery is itself a subcriterion of the Craft criterion in our research, and given its importance in other poetry research, we posit that it is one of the most important such factors.

- **Meter.** Each line is given a score between 0 and 1 based on its adherence to an iambic metrical scheme: for a score of 1, all even numbered syllables should be stressed and all odd numbered syllables should be unstressed. Because the stresses of single-syllable words can be difficult to discern, and because the CMU pronouncing dictionary also includes secondary stresses, our Meter scores are not exact. For metrical poetry, this is another obvious subset of Craft, and is necessary in order to produce poems of the desired form. The pre-selection of lines of the appropriate length and with appropriate rhymed endings also constitutes Craft.
- **Topicality.** Source texts are chosen for their general relevance to a specified topic. Each line is divided into trigrams based on a sliding window, and lines are given a higher score if they contain trigrams which occur more frequently in the data set. Early experiments with this version of TwitSong showed that the trigram frequency measure selected for common and intelligible turns of phrase and weeded out nonsensical combinations, but it often also resulted in the selection of bland lines which did not make it clear what the poem was about. So a specificity measure was added: the 30 most topical words in a given data file are selected by dividing the frequency of each word in the source text by its frequency in a non-topical comparison text (in this case, the comparison text is a compilation of poems from Poetry Magazine; a factor is added to the frequency to prevent division by zero). A trigram containing one or more of these most topical words receives a bonus to its topicality score. Topicality is necessary for the criterion of Meaning.

The four automated scores are then normalized and summed to give a line's total score. A RhymeSet is given its own score based on the average of the two top scoring lines in the set, and the top two lines of the highest scoring RhymeSets are arranged into a metrical rhyming poem by the poem construction mechanism.

TwitSong 3.0 uses sets of news articles as its source texts and assembles its lines into quatrains in Common Meter—an ABAB rhyme scheme with four iambs (eight syllables) in the A lines and three iambs (six syllables) in the B lines. This is the form of many hymns, including "Amazing Grace," as well as other popular poems and songs. This is different from how poems were constructed in previous versions of TwitSong. However, the major change in TwitSong 3.0 is the introduction of the Editorial Algorithm, by which the most promising lines can be refined by TwitSong after they are selected.

The Editorial Algorithm

The Editorial Algorithm is a form of genetic algorithm. However, instead of randomly recombining the most suc-

successful candidates in each generation—a technique which bears little resemblance to how human poets revise their work—we use a targeted edit at each step, replacing the words in each line that contribute most to the line’s worst-performing metric, out of the four metrics of Topicality, Emotion, Imagery, and Meter. We detail this algorithm below.

1. Initialization. The source text is read and the dictionaries used for each criterion are initialized, including the trigram frequency dictionary and identification of most topical words. We also initialize an interpolated Markov model (Salzberg et al. 1999) which can be used to generate additional text in the style of the source text. The Markov model can be up to order 3, but can flexibly reduce its order. If the model generates no results, or only one result, for a 3-gram, then it reduces the 3-gram to a 2-gram or 1-gram. The Markov model is trained to recognize punctuation that could indicate the end of a sentence or line, and runs until it generates an “end of line” marker; in an earlier version that did not use these markers, it was too common for a line to end on a preposition or other unsuitable word. Because the “end of line” marker is not guaranteed to occur after exactly 6 or 8 syllables, the Markov model in practice is run repeatedly until it generates a line that happens to be of the right length.
2. Line initialization. The source text is divided into lines of 6 or 8 syllables, separated by punctuation such as periods, question marks, colons, and commas. For each of the 30 most topical words, the Markov model generates a special line by starting with the listed word and iterating until it has a line with the appropriate number of syllables. Both the source lines and the Markov lines are then sorted into RhymeSets based on their end rhymes.
3. Scoring. The lines in each RhymeSet are scored based on the four metrics, and each RhymeSet is scored based on its best two lines. RhymeSets with a single line are scored, but penalized.
4. Trimming. For each RhymeSet, any line that is identical to the RhymeSet’s top scoring line, or that begins or ends with an identical word, is removed. Optionally, the programmer can also specify removal words that can only appear once in each RhymeSet; if the top scoring line contains one of these words, then any other line containing that word is removed. This is useful for preventing repetition. If more than 15 lines remain in the RhymeSet, it is then trimmed down to only its 15 highest scoring lines. The RhymeSets are re-scored and the 50 highest scoring RhymeSets are kept for the next generation, with RhymeSets of only a single line being removed first.
5. Edit planning. This is where the Editorial Algorithm identifies which words most need to be replaced. Each line in each RhymeSet is analyzed based on the four criteria. The criterion with the lowest normalized score, as well as any other criterion which is under a certain threshold, is selected for analysis. Each word in the line is then inspected for its contribution to this criterion, and the lowest performing word is selected for replacement. (“Stop words,”

such as “the” and “of,” are not excluded from this process; the thinking is that, if a stop word is present, there is no *a priori* reason why an alternate version of the line might not use a different sentence structure and have a higher-scoring word there instead.) For example, if Imagery is selected, then words that are very abstract are selected. We detail this process further below

6. Word replacement. The selected lines are sent to the Markov model which generates candidate replacement lines, starting with the selected underperforming word and replacing it and all subsequent words. (An earlier prototype of TwitSong replaced only the underperforming word, but this led to choppy and repetitive lines; an example is given in Table 1.) Because there is no guarantee that the replacement words will actually be better, the Markov chain generates many candidate replacement lines—20 for each selected starting word. These are then assigned to appropriate RhymeSets.
7. Successive generations. TwitSong repeats steps 3 through 6 to a maximum of 100 generations, or until the average score of the best ten lines stops increasing. In practice, the program very rarely runs for more than 15-20 generations, and sometimes as few as 3.
8. Poem construction. The top two lines each from the two highest scoring RhymeSets are selected. These are arranged into a quatrain in Common Meter.
9. Title generation. TwitSong generates a title for each of its poems, but the title generation mechanism is separate from the rest of the Editorial Algorithm. During the Line Initialization step, in addition to creating the initial RhymeSets, TwitSong also gathers a set of lines from the source text of 3 to 5 syllables without grouping them into RhymeSets. These potential title lines are then scored based on the four combined metrics and checked against the list of most topical words. Ideally, lines containing the first most topical word are selected and the highest scoring such line becomes the title. If there are no such lines, TwitSong will iterate down the list of most topical words. If no potential title line contains any of the 30 most topical words, TwitSong will choose the overall highest scoring potential title.

let wall street start off wall detroit’s
 and wall street start wall voiced
 let wall street start wall street wall point
 wall street out loud wall point

Figure 1: An early example of a poem from a prototype Editorial Algorithm, using Bernie Sanders’ lines from presidential debate transcripts as a source text. In this prototype, pairs of words were replaced during each edit. (An even earlier version, replacing single words, resulted in lines like “let wall wall wall wall wall wall street”.) This problem was avoided by a later protocol in which the target word and everything after it in the line is re-generated at once.

Source Texts

TwitSong 3.0's architecture allows it to generate poems quickly. In particular, the use of a Markov chain means that a relatively small source text can be used to generate poems. TwitSong 3.0's lower limit, before it stops being able to come up with sufficient numbers of rhymes for a quatrain, seems to be around 20 kilobytes of text. Therefore, it can be initialized with only a handful of articles on a breaking news topic.

We generated a great number of poems using TwitSong 3.0, mostly based on news articles from the BBC ¹, CBC ², Maclean's ³, and The Guardian ⁴. We chose these sources because they are mainstream, professional English language news sources which operate without a paywall. Occasionally we veered into other sources. For instance, when blockbuster movies were released, we collected fan responses to the movies from Tor.com ⁵ and The Mary Sue ⁶. We also tried alternative, non-news sources for some poems, such as classic novels available on Project Gutenberg ⁷.

The Evolutionary Algorithm in action

As an illustration, we show how the Evolutionary Algorithm uses its word replacement techniques on lines for a poem about the film *Avengers: Infinity War*.

One of the starting lines for this poem is:

thanos to grow the universe

This line receives high scores for meter and imagery, but a low score for topicality and a moderately low score for the chosen emotion, surprise. As both topicality and emotion are below their minimum thresholds, the Editorial Algorithm focuses on both of these.

Since topicality is calculated based on trigrams, TwitSong splits this line into its component trigrams:

thanos to grow / to grow the / grow the universe

The first and last trigrams are selected because they are not found in the trigram dictionary. Thus, TwitSong generates a set of candidate replacement lines starting at the beginning of the line, and a set of candidate replacement lines modifying only the last three words.

For emotion, TwitSong splits the line into its component words:

thanos / to / grow / the / universe

None of these individual words are very associated with the emotion of surprise, and some do not appear in the lexicon. Therefore, TwitSong flags all of them, and generates a maximal set of candidate replacement lines (a different set beginning the word replacement at each word).

The completed poem from this run of TwitSong reads:

¹<http://bbc.com/news>

²<http://www.cbc.ca/news>

³<http://www.macleans.ca>

⁴<https://www.theguardian.com/international>

⁵<https://www.tor.com/>

⁶<https://www.themarysue.com/>

⁷<http://www.gutenberg.org/>

Group A
FOR CANADA (<i>Olympics, joy</i>) hamelin pointing at the world team made it would be fair swiss stones for pavel is absurd swiss stones for him and there
Group B
WHY IS TRUMP SILENT (<i>Mueller investigation, disgust</i>) republican claims he will do flynn pleaded not care less committee has to look into pleaded not to the press
Group C
WAKANDA (<i>Black Panther, trust</i>) blackness as we love to her aid killmonger's plan to come conflict the atlantic slave trade sword and it was awesome

Figure 2: Example poems from the three experimental groups.

marvel had the fall of your mouth
luke of this journey through
infinity stone to point out
gags to where thor is too

Evaluation

Our goal in evaluating TwitSong was to falsifiably test whether or not the Editorial Algorithm and its associated line rating techniques improved TwitSong's poetry.

Method

We assembled three experimental groups of poems: Group A, Group B, and Group C.

Poems from Group A were generated according to the Editorial Algorithm described above. The best lines of each generation were edited with the goal of increasing their summed score on our criteria of Topicality, Emotion, Imagery, and Meter.

Poems from Group B were generated with a minimal version of the Editorial Algorithm. Lines were taken from a source text and generated based on a Markov chain trained on the source text. If this resulted in enough RhymeSets to produce a quatrain in Common Meter, the program was stopped there. Otherwise, it was allowed to iterate and perform the Editorial Algorithm for *only* enough generations to produce a valid quatrain. Every line was then assigned a score of zero, and the lines for the quatrain were chosen arbitrarily. Group B was meant as a control group in which the Editorial Algorithm did as little to improve the poems as possible, yet the poems were similar to the poems of Group A in every other respect.

We chose this method for our control group rather than using output from previous versions of TwitSong because

Emotion	Frequency	Topics
Disgust	7	Mueller investigation; the Parkland school shooting; Rex Tillerson; Doug Ford's election campaign in Ontario; March For Our Lives; the Stormy Daniels scandal; Viktor Orban's election in Hungary
Fear	6	Winter Olympics (2); Uber self-driving car crash; the Russian election; Austin bombing; NAFTA negotiations; Syrian chemical attack
Anticipation	5	Kim Jong Un's visit to China; Russian spy poisoning; US trade war; North Korea; Michael Cohen warrant
Anger	4	The Cambridge Analytica scandal; Facebook; Tim Hortons; Mark Zuckerberg
Joy	3	Winter Olympics (1); A Wrinkle in Time; Easter on April 1
Sadness	3	Stephen Hawking's death; Good Friday; Humboldt Broncos bus crash
Surprise	1	The Oscars
Trust	1	Black Panther

Table 1: Frequency of emotions from the NRC Hashtag Emotion Lexicon assigned to poems on different topics, from the group of 30 topics that were selected for the study. The topics in this table are sorted by associated emotion for ease of reading, and their order does not correspond to the ordering of topics in the study.

previous versions used different source text (Twitter) and a different metrical form; this is the first time that the Twit-Song system has been tested on news. Using a different baseline control group, such as random or human-generated text, would have given insight into where the poems stand in terms of overall quality, but would not have answered our specific experimental question about whether the Editorial Algorithm was improving the poems.

Poems from Group C were generated with a *reversed* Editorial Algorithm. That is to say, the line rating and edit planning steps were programmed to minimize instead of maximizing the poem's scores. So these poems were the Editorial Algorithm's attempt to make poems that were off-topic, unrelated to the selected emotion, abstract / devoid of imagery, and that failed to conform to an iambic stress pattern.

We chose a set of 30 news topics that were current at the time of the study and generated a Group A, Group B, and Group C poem for each. We then constructed a test set for our study in which, for each of the 30 topics, two of the groups were selected. The order of the news topics was not randomized, but the order of pairings (A vs B, A vs C, B vs A, B vs C, C vs A, or C vs B) was randomized across the set of news topics. All eight of the emotions from the NRC Hashtag Emotion Lexicon were present in our set of poems, but we made no attempt to balance or equalize the appearance of different emotions, instead picking the emotion that was most prevalent in articles describing each topic according to the NRC Hashtag Emotion Lexicon, with some normalization and some exceptions (see Table 1 for a full list).

We recruited experimental subjects by snowball sampling in order to include a reasonable number of poetry experts in our analysis; one of our authors is a published poet and recruited their own poetry contacts for the study. Each subject was directed to an online survey in which they were presented with each of the 30 pairs of poems and asked their opinions. Participants were also asked a few demographic questions and given a freeform text box at the end for other comments about the study. The full study took about 40 minutes and participants were given 10 Canadian dollars as

remuneration.

The bulk of the survey used a pairwise forced choice paradigm. For each pair of poems, participants were asked the following questions:

- Which poem do you prefer? (*General/Reaction*)
- Which poem is more creative? (*General*)
- Which poem does a better job expressing the emotion of [emotion]? (*Reaction*)
- Which poem does a better job describing the topic of [topic]? (*Meaning*)
- Which poem is more new and different? (*Novelty*)
- Which poem has better imagery? (*Craft*)

In a previous study (Lamb, Brown, and Clarke 2017) we included a question about cohesiveness. As TwitSong 3.0 does not contain mechanisms specifically designed to increase cohesiveness, we omitted this question from our study. Given the way many participants ended up focusing on the poems' lack of cohesion, this may have been a mistake.

Results

Demographics We divided our survey participants into experts and non-experts based on their self-reported experience with poetry. Experts were defined as participants whose poetry had been published in a magazine, anthology, collection, etc.

32 poetry experts participated in our study. This included 11 men, 10 women, 9 non-binary poets, and two experts who did not disclose their gender. (This is probably a serious overrepresentation of non-binary poets, but we do not expect it to affect our study results as none of the poems in the sample discuss queer/trans* issues.) Their ages ranged from 22 to 57, averaging 38. 28 of the 32 experts were native English speakers.

49 non-experts participated in our study, including 17 men, 27 women, and 5 non-binary participants. Their ages ranged from 17 to 64, averaging 32. 37 of the 49 non-experts were native English speakers.

We observed high attrition as the survey was rather long. Only 18 experts and 28 non-experts managed to complete every question. However, since the order of appearance of poems from different groups was randomized, this still left us with a good number of pairwise comparisons and did not present a major statistical problem.

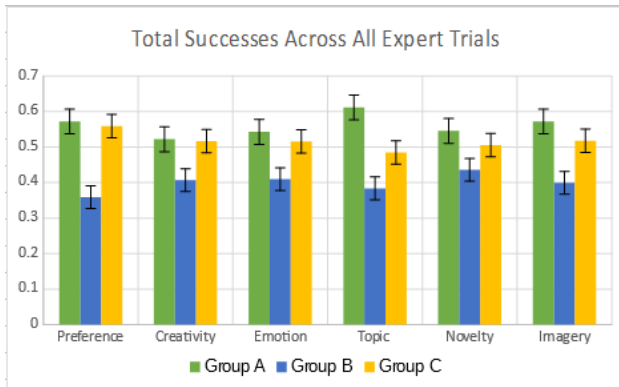


Figure 3: Success rates for types of computationally generated poems in pairwise comparisons with other poems, judged by experts. The height of a given bar represents the number of times a poem from that category was selected in preference to any other poem. The groups of bars add up to 150% because, for each category, the $\frac{1}{3}$ of trials in which a poem from that category does not appear are not considered. Error bars represent 95% confidence intervals, prior to Bonferroni correction.

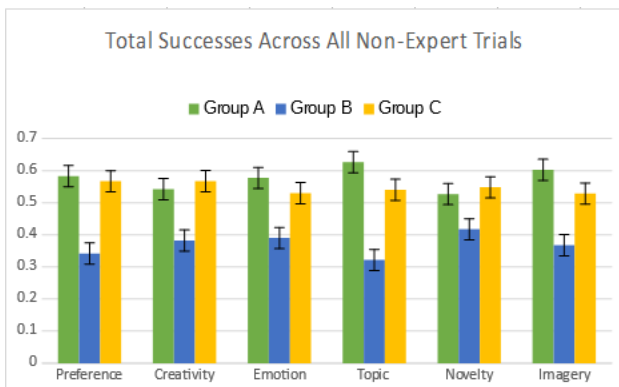


Figure 4: Success rates for types of computationally generated poems in pairwise comparisons with other poems, judged by non-experts. The height of a given bar represents the number of times a poem from that category was selected in preference to any other poem. Error bars represent 95% confidence intervals, prior to Bonferroni correction.

Group comparison We evaluated pairwise preferences between poems by treating them as a binomial distribution; statistical significance is calculated using the binomial theorem for cumulative probability. The null hypothesis is that

the probability of choosing a poem from one group over a poem from another, on any question, is 50%. As there are six questions, we applied a Bonferroni correction for multiple hypotheses, resulting in an alpha level of .0083 per test.

Our results are shown in Figures 3 and 4. As we had hoped, experts significantly preferred poems from Group A to poems from Group B on all six questions, $p < 0.0083$ for all. The differences between Groups B and C were not significant; surprisingly, neither were any differences between groups A and C.

Non-experts, like experts, significantly preferred poems from Group A to poems from Group B, $p < 0.0083$ for all questions. They also significantly preferred Group C to Group B on all questions, $p < 0.0083$. The differences between Groups A and C were not significant for non-experts.

Rather than the expected $A > B > C$ hierarchy, there is little difference between A and C. For experts there is some evidence of a possible $A > C > B$ ordering, but with the differences other than $A > B$ too slight to be significant. For non-experts, A and C seem to be genuinely statistically the same. This is illustrated in Figures 3 and 4.

Correlation between questions We looked at the correlations between the answers to our different questions, to see if our questions were truly capturing different dimensions underlying Product creativity. The results, for experts, are in Table 2. All the correlations between questions are above 0, which is not worrisome, since it is expected that a preference for a poem in some questions would have a priming effect on the other questions. However, some correlations are weak to moderate, while others are strong. There is a notable gap between the strongest moderate correlation (preference and emotion, Pearson's $r=0.56$) and the weakest strong correlation (creativity and imagery, $r=0.84$).

It appears that the measures of preference, creativity, novelty, and imagery are all strongly intercorrelated, while emotion and topic are more independent. This implies that experts evaluate the poems on three basic dimensions. One is how well the poem represents the target topic; another is how well the poem expresses the target emotion; a third is a more nebulous measure of how "good" the poem is, including novelty, imagery, and overall preference. Non-experts exhibited the same pattern as experts, with three underlying dimensions.

Freeform comments We counted and categorized the freeform comments made by experts and non-experts. Experts commented more often than non-experts, but there was more unity in the types of comments made by non-experts.

Several experts and non-experts stated that the poems didn't represent the intended emotions very well. Both experts and non-experts wished that there was a neutral/none/both option for times when neither poem met its targets well.

Experts were concerned about the poems' coherence. Several stated that the poems were incoherent, or that they cared more about coherence than the items the survey asked for. Two experts added that some lines were great, but that they were spoiled by proximity to incongruous or "word salad" lines.

	Preference	Creativity	Emotion	Topic	Novelty	Imagery
Preference	1					
Creativity	0.855	1				
Emotion	0.563	0.360	1			
Topic	0.525	0.320	0.344	1		
Novelty	0.839	0.861	0.351	0.227	1	
Imagery	0.904	0.837	0.421	0.423	0.892	1

Table 2: Correlations (Pearson’s R) between answers to each of the six questions, as judged by experts.

Non-experts made more comments about the overall quality of the poems, although they were divided in their responses. Several said that the overall set of poems, or the idea for the study, was interesting or cool. Some said that the poems overall are not very good, while others said that some individual poems were quite good. Several non-experts indicated that the poems were hard to understand or didn’t make sense, which may be the non-expert version of complaints about coherence.

One expert commented, “My god, that was awful. The poems were some of the worst computer-generated texts I’ve ever seen.” In contrast, a non-expert said, “This is a really interesting study—I was trying to guess which poems were computer-generated as I did the survey, and I couldn’t tell most of the time!” This comment is notable since we had intended it to be clear that *all* poems in the study were computer-generated.

Discussion

We were surprised by our results. It seems that the Evolutionary Algorithm improves poems even when told to make the poems worse.

We can think of a few ways to interpret this result. One is that our line rating metrics are useless and something else about the Evolutionary Algorithm improves the poems. However, we are not sure what that would be. Although participants claimed not to see much difference between the groups, they detected a statistically significant difference. This difference must be due to the line rating metrics and their use, as there were no other consistent differences between the poems from the three groups.

It is possible that, while the line rating metrics are useful, their reverse versions are also useful. This is most easily explained with Meter. A line with a score of 1.0 for meter is a perfect iambic line. However, the opposite of an iambic line is not an unmetrical line. Instead, the opposite of an iambic line is a trochaic line. It is very likely that, while lines from Group B had random stress patterns and lines from Group A were mostly iambic, lines from Group C were mostly trochaic. Looking at the poems from group C, many do contain trochaic or close to trochaic meter, with lines like *game that finish gave a doping or shooting following his thursday*.

This explanation is speculative due to a flaw in our experiment: we did not include a question like “Which poem has better meter and rhythm?” even though rhythm and meter are valid subcategories of Craft.

If Groups A and C have good meter and Group B does not, then there are two possible explanations for the other results. One is that the answers to the other questions are illusions—survey participants prefer the poems with better meter, and this increases scores in other areas solely due to priming. Another possible explanation is that other line rating metrics also exhibit this reverse effect. A poem with low Topicality might contain more unusual trigrams and, thus, more Novelty. A poem with a low rating for one emotion might end up exhibiting another, equally interesting emotion. Lines with lower Imagery might use more straightforward language and therefore be more coherent. This explanation does not completely explain the data; for instance, it does not explain why Groups A and C are both more topical and more novel than Group B.

We suspect a combination of both explanations. Both Group A and C improve on the Group B poems, especially for meter, but Group A is slightly more on target with regards to its other goals. Experts are more sensitive to this, resulting in a ranking where Group A (slightly, non-significantly, but consistently) outperforms Group C, while non-experts are more fully swayed by meter and less able to perceive other improvements. Although the difference between Groups A and C when judged by experts is not significant, there is only a 1/64 chance that Group A would outperform Group C on all six questions if the data was random.

If this combined explanation is true then we would expect several consequences in further experiments. First, we would expect that, if we did include a question about Meter, Group A and C would prove to have better meter than Group B, and other questions would be highly correlated with Meter, especially for non-experts. Second, if we had a better implementation of our line ratings, then the difference between Group A and Group C, at least for experts, would increase.

Conclusion

TwitSong 3.0 was meant to build on the accomplishments of WASP (Gervás 2013a; Gervás 2013b; Gervás 2016). By making goal-directed edits to candidate lines as WASP does, TwitSong 3.0 measurably improves these lines. However, our goal was to expand these techniques to semantic goals such as topicality and emotion, and our success at this was limited: the improvements that our system made to its lines were mostly in the area of meter. Editing lines to be more topical remains an open problem. Additionally, TwitSong 3.0’s poems were generally not very coherent or well liked

by expert judges.

While TwitSong 3.0 has mixed success as a poetry system, it also exemplifies the importance of well-constructed evaluation with expert judges and falsifiable hypotheses. Without such testing, we might have sensed that the generated poetry wasn't as good as we wanted, but we would not have had the detailed statistical insight that helped us figure out the reason for this and to discover one part of the system (meter) that was working well. There is room to improve our evaluation techniques further, for example, by testing and standardizing a more robust set of questions.

References

- [Astigarraga et al. 2017] Astigarraga, A.; Martínez-Otzeta, J. M.; Rodríguez, I. R.; Sierra, B.; and Lazkano, E. 2017. Poet's Little Helper: a methodology for computer-based poetry generation. A case study for the Basque language. In *Proceedings of the Workshop on Computational Creativity in Natural Language Generation (CC-NLG 2017)*, 2–10.
- [Dahlstedt 2012] Dahlstedt, P. 2012. Between material and ideas: a process-based spatial model of artistic creativity. In *Computers and Creativity*. Berlin: Springer. 205–233.
- [Díaz-Agudo, Gervás, and González-Calero 2002] Díaz-Agudo, B.; Gervás, P.; and González-Calero, P. A. 2002. Poetry generation in COLIBRI. In *Advances in Case-Based Reasoning*. Springer. 73–87.
- [García et al. 2006] García, R.; Gervás, P.; Hervás, R.; Pérez, R.; and ArÁmbula, F. 2006. A framework for the ER computational creativity model. In *MICAI 2006: Advances in Artificial Intelligence*, 70–80. Apizaco, Mexico: Springer.
- [Gervás 2013a] Gervás, P. 2013a. Computational modelling of poetry generation. In *Artificial Intelligence and Poetry Symposium, AISB Convention*. Exeter University, UK: Society for the Study of Artificial Intelligence and the Simulation of Behaviour.
- [Gervás 2013b] Gervás, P. 2013b. Evolutionary elaboration of daily news as a poetic stanza. In *Proceedings of the IX Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados-MAEB*, 229–238.
- [Gervás 2016] Gervás, P. 2016. Constrained creation of poetic forms during theme-driven exploration of a domain defined by an n-gram model. *Connection Science* 28(2):111–130.
- [Hirjee and Brown 2010] Hirjee, H., and Brown, D. G. 2010. Using automated rhyme detection to characterize rhyming style in rap music. *Empirical Musicology Review*.
- [Kao and Jurafsky 2012] Kao, J., and Jurafsky, D. 2012. A computational analysis of style, affect, and imagery in contemporary poetry. In *NAACL Workshop on Computational Linguistics for Literature*, 8–17.
- [Lamb, Brown, and Clarke 2015] Lamb, C. E.; Brown, D. G.; and Clarke, C. L. 2015. Can human assistance improve a computational poet? *Proceedings of Bridges 2015: Mathematics, Music, Art, Architecture, Culture* 37–44.
- [Lamb, Brown, and Clarke 2016] Lamb, C.; Brown, D. G.; and Clarke, C. L. 2016. Evaluating digital poetry: Insights from the CAT. In *Proceedings of the Seventh International Conference on Computational Creativity*. Association for Computational Creativity.
- [Lamb, Brown, and Clarke 2017] Lamb, C.; Brown, D.; and Clarke, C. 2017. Incorporating novelty, meaning, reaction and craft into computational poetry: a negative experimental result. In *Proceedings of the Eighth International Conference on Computational Creativity*. Association for Computational Creativity.
- [Lamb, Brown, and Clarke 2018] Lamb, C.; Brown, D. G.; and Clarke, C. L. 2018. Evaluating computational creativity: An interdisciplinary tutorial. *ACM Computing Surveys (CSUR)* 51(2):28.
- [Malmi et al. 2015] Malmi, E.; Takala, P.; Toivonen, H.; Raiko, T.; and Gionis, A. 2015. DopeLearning: a computational approach to rap lyrics generation. *arXiv preprint arXiv:1505.04771*.
- [Mohammad and Kiritchenko 2015] Mohammad, S. M., and Kiritchenko, S. 2015. Using hashtags to capture fine emotion categories from tweets. *Computational Intelligence* 31(2):301–326. Full lexicon available at <http://saifmohammad.com/WebPages/lexicons.html>.
- [Provalis 1990] Provalis. 1990. Regressive imagery dictionary. <https://provalisresearch.com/products/content-analysis-software/wordstat-dictionary/regressive-imagery-dictionary/>.
- [Rashel and Manurung 2014] Rashel, F., and Manurung, R. 2014. Pemuisi: a constraint satisfaction-based generator of topical Indonesian poetry. In *Proceedings of the Fifth International Conference on Computational Creativity*, 82–90. Ljubljana, Slovenia: Association for Computational Creativity.
- [Salzberg et al. 1999] Salzberg, S. L.; Perteau, M.; Delcher, A. L.; Gardner, M. J.; and Tettelin, H. 1999. Interpolated Markov models for eukaryotic gene finding. *Genomics* 59(1):24–31.
- [Simonton 1990] Simonton, D. K. 1990. Lexical choices and aesthetic success: A computer content analysis of 154 Shakespeare sonnets. *Computers and the Humanities* 24(4):251–264.
- [Simonton 2011] Simonton, D. K. 2011. Creativity and discovery as blind variation: Campbell's (1960) BVS model after the half-century mark. *Review of General Psychology* 15(2):158.
- [Toivanen, Gross, and Toivonen 2014] Toivanen, J. M.; Gross, O.; and Toivonen, H. 2014. The officer is taller than you, who race yourself! Using document specific word associations in poetry generation. In *Proceedings of the Fifth International Conference on Computational Creativity*, 355–359. Association for Computational Creativity.
- [Ward, Smith, and Finke 1999] Ward, T. B.; Smith, S. M.; and Finke, R. A. 1999. Creative cognition. In *Handbook of creativity*. Cambridge, UK: Cambridge University Press. 189–212.

Evolving the INES Story Generation System: From Single to Multiple Plot Lines

Eugenio Concepción¹ and Pablo Gervás^{1,2} and Gonzalo Méndez^{1,2}

¹Facultad de Informática

²Instituto de Tecnología del Conocimiento
Universidad Complutense de Madrid

{econcepc, pgervas, gmendez}@ucm.es

Abstract

INES (*Interactive Narrative Emotional Storyteller*) is a story generation system based on the Afanasyev framework. It is focused on generating stories by combining template-based plot generation with an agent-based simulation of characters' interaction. Its design follows the microservice-oriented model established by Afanasyev, in which a Story Director orchestrates the story generation stages, implemented by specialised microservices. While this model is suitable for generating a single plot story, it is insufficient for managing a multiple plots scenario. This paper focuses on describing an evolved version of INES that aims at generating stories that contain different plot lines. In addition to the adoption of changes in the story representation model, the adaptation entails a modification of the operation of INES and includes a new microservice: the Plot Weaver. This component introduces the application of a literary technique referred to as the “*Communicating Vessels*”, in which the different lines evolve in parallel while interacting between themselves.

Introduction

Automated story generation is a research area in Artificial Intelligence focused on developing systems whose result is a story (Gervás 2012). It is closely related to other Computational Creativity areas such as interactive storytelling (Glassner 2009).

Story generation systems present two distinctive characteristics: they strongly depend on knowledge and they can only generate a constrained variety of stories (Gervás and León 2014). A story generator requires knowledge from practically all areas of the collective wisdom, so it needs to be fed with a wide range of information, from the most basic common sense knowledge to the physical world rules. Besides this, due to the technical limitations of the generation process, story generators only generate stories of a certain kind –in terms of theme, rhythm, discourse, etc. Many of these limitations come from the fact that the architecture of these systems is built according to a monolithic design. Hence, a single application concentrates all the required functionality and assets. If this is combined with the lack of architectural mechanisms for collaborating with other systems, the results obtained are quite restrictive. A way of addressing all these limitations is by adopting a distributed architecture, with an emphasis on the collaboration between different systems.

INES (Concepción, Gervás, and Méndez 2018b) is a story generation system based on Afanasyev, a microservice-oriented architectural framework (Concepción, Gervás, and Méndez 2018a). Afanasyev provides a reference architecture that brings a collaborative environment for services sourced from different storytelling systems and orchestrated by a Story Director. INES was originally developed to test the suitability of the framework and also as an evolution of the Charade storytelling system (Méndez, Gervás, and León 2016). The limitation of the current version of INES comes from the fact that it can only generate a single-plot story, being its design inadequate for managing a multiple plots scenario.

Background

TALE-SPIN (Meehan 1977), one of the first story generators, wrote short stories about the inhabitants of a forest. From a technical point of view, it applied planning techniques (Cohen and Feigenbaum 2014) for generating the characters actions –while trying to achieve their goals, and then it wrote up the story by narrating the steps performed by the characters for achieving their goals. **Author** (Dehn 1981) was also a planner but focused on the authorial goals instead of the characters' goals. **Universe** (Lebowitz 1984) generated scripts for a TV soap opera by focusing on characters interaction. Its generation process included a planning stage that kept track of pending goals for developing a partial draft of the story until plot completion. **GESTER**, *Generating STories from Epic Rules*, (Pemberton 1989) was one of the first approaches towards generating stories from interacting modules of independent knowledge. The program was a rule-based story generation system that managed information about story structure, in the form of a simplified version of a narrative grammar, and to the possible events and actors of the epic sub-genre. **Brutus** (Bringsjord and Ferrucci 1999) generated short stories about betrayal. It used a very thorough knowledge model for representing the concept and implication of betrayal. It also provided a grammar-based generation model and a literary beautifier, which allowed it to generate high-quality stories, providing texts that could have been written by humans. **MEXICA** (Perez y Perez 1999) was a storytelling system that generated mythological stories about the Mexicas, the early inhabitants of Mexico. It was the first system that brought the character's

emotions into play in the generation process. **MAKEBELIEVE** (Liu and Singh 2002) generated short fictional stories using common sense knowledge to generate them. It required the user to provide a story about a character as initial seed, for later attempting to continue that story by inferring possible sequences of events that might happen to that character. **TEATRIX** (Machado, Paiva, and Brna 2001; Prada, Machado, and Paiva 2000) was a virtual environment for story creation, designed to help children and teachers to understand the whole process of collaborative story creation. It provides an environment where both drama and story creation are merged into one medium. Architecturally, it is an agent-based system in which each character is performed by an intelligent software agent interacting in the story world. Every character plays a role according to the Propp's folktales model (Propp 1968). **Fabulist** (Riedl and Young 2010) is a whole architecture for automatic story generation and presentation which combines both the author interests and the characters intentionality. **Charade** (Méndez, Gervás, and León 2014; 2016) is a storytelling system focused on the relationships between the characters. By simulating their interactions, it tracks the evolution of their mutual affinities and applies it for generating stories. From an architectural point of view, it is an agent-based architecture implemented in JADE (Java Agent Development Framework) (Bellifemine, Poggi, and Rimassa 1999). Charade aims at implementing an affinity model decoupled from the story domain, that is, the world in which the story takes place or any other context-related attribute of the characters.

Related work

Single-plot stories are not the only model in human-made narrative. There are a good number of stories that contain more than one plot, such as unnatural narratives, a subset of fictional narratives that subvert the physical laws, the logic principles or the standard human limitations (Alber and Heinze 2011). This approach affects not only the facts told in the story, but also its structure: it can contain several interwoven plots, a rupture of the plot natural progression, multiple concurrent plots and many other possibilities. From the beginning of the literature, the unnatural elements have been present in the literary production (Todorov 1975).

In addition to the historical precedents, postmodern literature has adopted much of those unnatural resources, bringing a disruptive narrative design to the stories (Martínez 2011). Contemporary literature is fraught with stories containing several plot lines linked together by means of diverse strategies (Menéndez 2013).

There is not a long record of multiple plot generation and combination in automatic story generation. One of the remarkable works in this respect is the plot weaving algorithm proposed by Fay (Fay 2014). This is a method that takes a set of individual plot threads as input, one for every single character, and generates a new story by tying them. These individual plot threads have been previously extracted from a preexisting story by means of the *Genesis* story understanding system (Winston 2016). This plot weaving algorithm makes sure that characters' plots are compatible and it also takes care of building a consistent timeline for all of the plot

elements of the story. This procedure is computationally difficult because it entails selecting the best set of pairings of characters to generic entities to create the best possible combination for the story (Fay 2014).

Porteous et al. (Porteous, Charles, and Cavazza 2016) have developed a remarkable example of multiplot interactive storytelling system. It is focused primarily on facing three challenges: the distribution of the characters across the different subplots, the length of each subplot presentation and the transitions between subplots.

Gervás (Gervás 2018) has recently explored the suitability of combining events from a sequence for generating a plot as a technique for story generation. This approach has partially influenced the solution proposed in this paper.

It is also worth mentioning other efforts in the generation of multiple possibilities in a story, such as the planning approach by Li and Riedl (Li and Riedl 2010) and the *Crystal Island* (Mott, Lee, and Lester 2006) interactive narrative engine. In the first case (Li and Riedl 2010), authors define a plan refinement technique based on partial-order planning that it uses for off-line adaptation of authored narratives with multiple "quests" adapting the plot line to create new plausible sequences of actions. The Crystal Island generation model offers multiple quest subplots that encourage the user goal recognition, combining multiple plot elements to create rich customized stories (Mott, Lee, and Lester 2006). However, whilst these approaches sought to generate multiple quests, they did not provide a procedure to interleave them.

Materials and methods

This section covers the technical basis and the conceptual techniques considered for designing the solution. It firstly provides a review of INES, with emphasis on those aspects that mainly take part in the plot generation step, and some known literary techniques for creating stories based on multiple plots. It also focuses on reviewing the knowledge representation model used by INES, taken from the Afanasyev common knowledge representation structure. Lastly, it describes different literary strategies for writing multiple plot narratives.

The INES story generation system

INES (Interactive Narrative Emotional Storyteller) is based on the Afanasyev framework (Concepción, Gervás, and Méndez 2018a; 2018b). One of the declared objectives of this framework is to facilitate the generation of stories much closer to human-made literature by combining the diverse capabilities of various story generators (Concepción, Gervás, and Méndez 2018a). Afanasyev provides both a microservice-oriented reference architecture and a common knowledge representation model.

The architecture of INES is based on microservices. Figure 1 depicts a high-level view of its original components. The following lines focus on a short review of the relevant aspects of the architecture. A more detailed description can be found in (Concepción, Gervás, and Méndez 2018b).

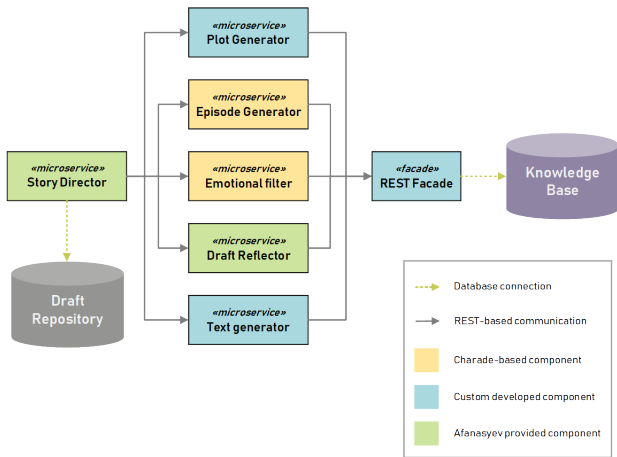


Figure 1: Original architecture of INES.

The operation of this microservices ecosystem is driven by the Story Director. It orchestrates the rest of the services to perform the story generation process. Story Director starts by requesting the Plot Generator to generate a plot. This request includes a list of initial characters, the plot template to apply and the setting in which the story happens. Every plot is a sequence of scenes characterised by a precondition and a postcondition that reflect the state of the world before and after the development of each scene (Concepción, Gervás, and Méndez 2018b; 2018a).

The knowledge representation model provided by Afanasyev tries to cover all the aspects related to the structure and meaning of a story (Concepción, Gervás, and Méndez 2017). This representation has been designed as a hierarchical structure in which the root concept is the **story**. A story represents what both intuitively and narratologically can be considered a story, that is, a narration of events happening in a setting (Concepción, Gervás, and Méndez 2017). It is composed by the two classic narratological components: the plot and the space.

The plot of the story represents its essential structure, providing a sort of scaffolding for the actions and events that happen across the story. In the Afanasyev model, the plot is generated at the beginning by the Plot Generator microservice.

The INES instance for the **Plot Generator** is named “**Audrey**” —after Audrey Hepburn. It is a template-based plot generator which produces outlines from a subset of the cinematographic basic plots compiled by Balló (Balló and Pérez 2007). Audrey aims at building a story plot containing the main scenes that will be completed by the Episode Generator microservice. Its generation model is quite similar to systems like Gester (Pemberton 1989) and Teatrix (Machado, Paiva, and Brna 2001).

The original plot building procedure started by selecting one of the cinematographic templates, namely a conceptual structure with a sketch of the plot, and developed it later by instantiating the roles and the types of actions into real char-

Episode	Description
Initial state	A peaceful community
Arrival	The arrival of the outsider to the community
Outsider destructive actions	The outsider acts against the members of the community, performing destructive actions, without being uncovered
The outsider revealed	The true evil nature of the outsider is revealed
The rise of the heroes	The heroes rise from the community and fight against the outsider
Purge	The outsider is purged. The community becomes peaceful again

Table 1: The “Destructive Outsider” story plot template

acters and actions. Audrey’s REST interface supports the random selection of a template but also the selection of a specific template name. Once a basic template is selected, it instantiates its generic elements to develop a concrete plot. This step entails the use of knowledge about the context in which the story will be set. In this case, the context is inferred from the request parameters.

An example of one of these templates is “*The destructive outsider*”, summarized in table 1 (Concepción, Gervás, and Méndez 2018b).

Audrey queries the knowledge base that contains the main concepts presented in the plot for creating a consistent detail for every episode (Concepción, Gervás, and Méndez 2018b) according to certain setting. In the prior example, the plot mentions a “community”, an “outsider”, some “destructive actions” performed by the outsider, etc. All these concepts are included in the knowledge base and there are more specific roles and actions which refer to them. For example, the concept of “community” can be instantiated into a “town”, a “family” or a “company”. In each case, the “outsider” can be a “new sheriff”, an “unknown relative” or a “new colleague”. By the same token, the knowledge base contains the information required to determine the type of actions that the characters can perform.

Every episode or scene is expressed in terms of a set of attributes that essentially provide information about the story space—including both time and location, the characters that appear in the episode and the state of the world before and after the episode happens (Concepción, Gervás, and Méndez 2018a; 2018b). These last information is represented as the scene precondition and postcondition. They are sets of assertions expressing the restrictions to be considered during the development of the episode’s detail. Table 2 shows a sample of all these attributes.

Probably, the most influential attributes in terms of consistency keeping across the scenes are the precondition and the postcondition. They are included as a part of the Scene - Frame - State structure. These attributes are a collection of assertions about the state of the story world before and af-

Attribute	Value
Precondition	John is a friend of William
Postcondition	John is a friend of William John performs friendly actions to William
Characters	John, William
Time	Story relative time in which the episode happens
Location	Spatial reference in the story world

Table 2: The basic attributes of an episode

ter the scene occurs. They contain assertions related to any existant referenced in the scene—that is, characters, beings and objects.

Techniques for plots interweaving

This section reviews some feasible techniques for plot interweaving taken from the Literary and Narratological studies.

The “*Chinese Box*” technique (Menéndez 2013) consists in the inclusion of nested stories inside a larger one. Every nested story can be related to one or several characters of the main plot line, or even consist of a separate one with the purpose of explaining some happenings of the main story. This approach has been applied in great works of literature such as *The Arabian Nights* (Vernet 1990) and *Don Quixote* (Cervantes 2011).

The technique of the “*Communicating Vessels*” (Menéndez 2013) is based on constructing a story by alternating at least two differentiated parallel plot lines. An example of this technique can be seen in *Madame Bovary* (Flaubert 1857), which contains a chapter that alternates two apparently disconnected plot lines. The only commonality between them is the temporal context in which the actions and happenings they contain are occurring. The resulting effect is a contamination between the two plot lines that, taken in isolation, would produce a different understanding in the reader. In other words, when two alternating story passages are interwoven together, proximity and alternation generate mutual influence. This influence can be applied to the tone, the tension, or the atmosphere that every story transmits to the other.

However, this technique is not limited to these types of influence. A stronger influence is also possible at the plot level. That means that both lines can develop a closer connection and converge at a certain point in the story. This can be achieved by using shared characters in both plots, acting as a hook between them.

Regardless of how strongly connected the plot lines are, it is important that there is a balance between the two plots, in order to avoid that one predominate over the other.

Despite the prior explanation has focused on the application of the technique on only two plots, it is also applicable to more than two plot lines. An example of several overlapping plots in which the characters intersect is *Pulp Fiction* (Tarantino 1994).

Proposed solution

This section focuses on detailing the approach adopted for implementing the multiple plot interweaving techniques described above and the consequent evolution of the existing architecture of INES, putting special stress on the new central component of the solution: the Plot Weaver service.

Multiple-plot generation process

The structure of a story plot in INES is based on a sequence of scenes, each of which is defined by a set of preconditions, a set of postconditions, the time in which it happens and a spatial reference in the story space (Concepción, Gervás, and Méndez 2018a; 2018b). According to this model, the weaving of scenes from different plots should take into consideration the logical consistency when combining their respective pre and postconditions. Following this reasoning, there are two ways of weaving scenes from different plots: combination and juxtaposition. The first strategy combines two scenes from different plots into a new one. This operation is feasible if and only if both the preconditions and the postconditions from the two scenes are respectively consistent among themselves from a logical point of view. The juxtaposition approach creates a combined sequence of scenes by putting one after another. In this case, the postcondition of every scene must be consistent with the precondition of the scene that goes after it. The latter is the approach adopted for the design of the Plot Weaver. This microservice, that will be concisely described later, is responsible for implementing the interweaving of the plot lines.

So, in order to weave the plots, the preconditions and the postconditions of the involved episodes must be consistent. If not, the Plot Weaver skips one episode in the plot line and tries to match with the next one. There is always a chance that the plot lines are simply incompatible so the response in this case would be an error, and the Story Director would have to select a different pair of plots to merge.

The new generation process starts with the Story Director requesting the Plot Generator to create a first plot. The request includes as parameters a template, the characters’ list and a theme. The *template* parameter is the logical name which references the plot template that must be instantiated to create the plot. This instantiation is strongly linked with the *theme* parameter, that represents the setting in which the story will take place. It is also a logical name referencing a particular context in the knowledge base. For example, when applying a plot template such as “The Destructive Outsider” summarised in Table 1, the “Community” can be a “Nineteenth Century Western North American Town” or a “Middle-Class Family”, depending on whether the theme is “Far West” or “Family Drama”.

The generated plot is the skeleton of a story draft that is persisted in the Draft Repository. This draft contains information about the setting—time and space in the story world—, a list of the characters mapped with the roles required by the template and the theme, and of course the plot line.

Following this first plot generation, the Story Director requests the Plot Generator to create a second plot. This time,

the request includes as parameters a new characters' list containing a subset of the characters' list of the first plot, their roles, the setting —time and space—, and the theme of the existing draft. In addition, the Plot Generator will need to locate a proper template which can fit the character roles required, according to the theme. In order to do this, it must query the template repository to get all the available templates that consider the involved roles. If no match is found, then the characters' list will not contain any common character with the prior plot line. This case will produce a story with two parallel plot lines that take place in the same setting, but apparently unrelated.

The procedure described above shows how the Story Director keeps the global consistency between the two generated stories. After having generated the two plots, the weaving process can start. The Story Director requests the Plot Weaver to merge the plots of the two generated stories.

The weaving of the plots is performed according to a sequence of steps. Taking the plot of the first story as the master line, the Plot Weaver proceeds by finding a compatible scene in the second plot line and merging it with the first one to insert a new scene in the master plot line. This means that the weaving process always takes the sequence of episodes of this first plot and proceeds by looking for compatible episodes in the second plot. The ideal outcome of this procedure is the generation of an interwoven plot line consisting in an alternation of scenes from the first and the second plot. The limitation of this strategy is that non-merged scenes from the source plots could remain, which would be included at the end of the resulting plot.

Evolved INES architecture

Adding the objective of supporting multiple-plots generation to the current operational requirements, the original generation model is insufficient. This need adds a new functional driver to the existing architecture. The adaptation of INES for interweaving plot lines is based on the definition of a set of plot merging heuristics and the modification of the INES microservices ecosystem to include this stage in the generation process. Figure 2 depicts the current architecture after evolving INES to support multiple plot generation.

The original INES model considered only one plot per story. This entails that the Story Director only requests the Plot Generator for a plot line once for every story. In the evolved version, the Story Director requests twice the Plot Generator to get the plot lines to merge and must include new controls along its inner logic to prevent inconsistencies.

As described previously, the original REST interface of the Plot Generator supported requests with no parameters, so it instantiated a random template, and requests with the template to be applied to generate the plot. In this case, generating a consistent story requires that the different plots share a common story space. This entails not only location and time, but also the set of participating characters. So that, the REST interface of Audrey —the Plot Generator, has been modified to accept all these new parameters. In order to adapt the architecture for mixing two plot lines into a single story, it is also necessary to include a new component. However, the adaptation of this microservice has entailed more changes

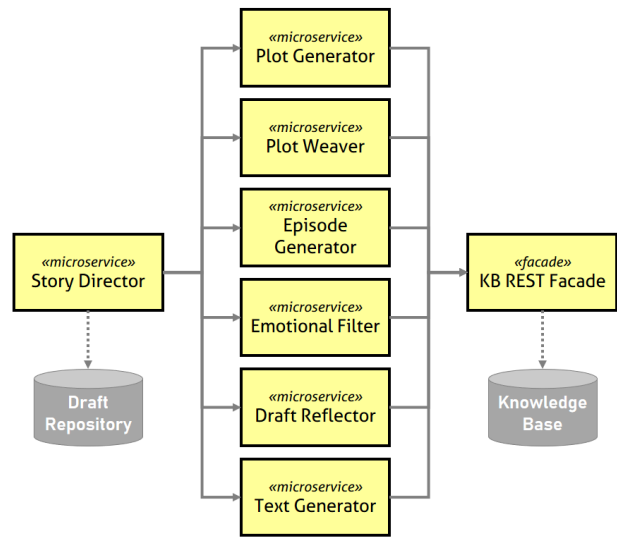


Figure 2: The evolved architecture of INES.

than merely adapting its interface. Audrey uses a inner template repository for managing the templates it instantiates during the plot generation. The original design of this component only allowed for querying templates by name. Due to the need for having a way of selecting templates by the roles they involve, the signature of the template repository has been adapted to provide queries based on roles. Thus, the updated Plot Generator can choose a template to apply according to certain restrictions. This functionality is essential to implement the multiple plot generation procedure.

The Plot Weaver is the microservice devoted to perform the plot weaving stage. It is implemented according to the Strategy pattern in order to select and apply the selected weaving heuristic. Initially, there have been considered two heuristics derived from the “Communicating Vessels” technique. The simplest way of mixing the plots is by a mere alternation of episodes, a kind of “unrelated juxtaposition” of episodes from the two plot lines. In this basic case, the sets of characters of the two original plots can be disjointed. A more elaborated way is the linking of the episodes according to their compatibility in terms of state of the story world. In this case, there are common characters among the two stories.

The Plot Weaver provide as the default strategy the alternation of episodes from the two plots to combine —by juxtaposition. The resulting plot line will be a sequence of episodes picked from the two initial plots. Beside this, it also provides the option of merging the plots by sharing a subset of every plot’s characters. This operation is the most complex and requires the Story Director to share certain parameters during the plot generation stage. So, it has to request the Plot Generator twice, for generating the two plots to combine, and provide shared information as parameters:

- Setting reference, as mentioned before, in order to instantiate the plot template according to a particular setting in the knowledge base, the Plot Generator requires a refer-

Episode	Description
Initial state	A frustrated character regrets his / her fruitless life
Temptation	The character is tempted by another character representing the evil forces
Pact with evil	The main character agree to serve the evil cause in exchange for a new satisfactory way of life
Evil actions	The main character performs evil actions induced by evil
Enlightenment	The main character becomes aware of being enslaved by evil
Redemption	The main character performs a saving action and dies. Evil is defeated

Table 3: The “Faust” story plot template

ence for this setting —e.g. “Far West”, “Present day”, “Epic Fantasy”.

- Characters list, containing their name and their role in the plot template. The Story Director has to select which characters will be shared between the two plots in order to request the Plot Generator to include them.

In addition, the Story Director must take care of not requesting the Plot Generator using the same theme —e.g. requesting to generate two plots for interweaving based on “The destructive outsider”, and choosing a compatible plot template for the second plot.

The interface of the Plot Weaver is also a stateless REST-based API, as the rest of the microservices of INES. In a first version, the Plot Weaver is only capable of combining two plot lines. An obvious precondition for these two plots to be merged is that they share a subset of the characters involved in their respective plot lines. For assuring this precondition, the Story Director must analyse the characters’ roles of the first plot before requesting the Plot Generator for the second time. It needs to identify a set of matching roles between the first plot line and any of the available plot templates for the second plot line. For this reason, the REST interface of the Plot Generator includes a new operation for requesting information about the available plot templates and their metadata —such as characters’ roles.

An example of interwoven plot story

The following lines introduce an example of story generation by plot interweaving according to the “Communicating Vessels” technique. It is structured around a first plot based on the “Destructive Outsider” template and a second one based on the “Faust” template. Tables 1 and 3 show the detail of both plot templates.

Table 4 shows a sample story which combines the two plots, generated by applying a juxtaposition approach with shared characters. The story combines two plots based on two different templates. The white rows contains the

Episode	Actions
A peaceful community	Mary and John work together on their farm William helps John with the farm tasks Mary invites William to dinner Jeff visits John Jeff gives a present to John
Frustrated character regrets his life	Jeff feels miserable Jeff thinks that he is weak Jeff hates Carlson Jeff wants to arrest Carlson
The arrival of the outsider	Adam arrives at the city Adam buys a ranch Jeff welcomes Adam John welcomes Adam Mary invites Adam to dinner
Temptation	Adam offers help to Jeff Adam offers money to Jeff Adam tells Jeff to arrest all the gunmen
Outsider destructive actions	Adam wants John’s farm Adam sneakily burns down John’s barn
Pact with evil	Adam blames Carlson for burning down John’s barn Jeff accepts Adams’ money Jeff arrests Carlson
Conflict	Adam offers Mary to buy her farm Mary accepts Adam’s offer John refuses to sell his farm John gets angry with Mary
Evil actions	Adam shoots John John is injured
The outsider revealed	Mary witnesses Adam shooting John Mary tells Jeff that Adam is a killer John tells Jeff that Adam is a killer Jeff gets angry with Adam
Enlightenment	Jeff realises that Carlson did not burn out John’s barn Jeff releases Carlson Jeff says sorry to Carlson
The rise of the heroes	John faces Adam John demands Adam to leave Adam refuses to leave the town
Redemption	Jeff arrests Adam Adam shoots Jeff Carlson shoots Adam Adam dies Jeff dies
Conclusion	Mary says sorry to John Carlson is freed

Table 4: A sample story based on mixing two plots

episodes from the first plot—based on “The destructive outsider” template, while the gray rows contains the episodes from the plot based on “Faust”. Both plots share the same setting and a subset of the characters. The story takes place in the Far West and the characters of the whole story are the following:

- John, a farm owner married with Mary
- Mary, John’s wife
- William, a farmer friend of Mary and John
- Jeff, the Sheriff
- Carlson, a gunman
- Adam, a cattle baron, an outsider

William is a character that only appears in the first plot, while Carlson only appears in the second one. Despite this, the combined plot line remains consistent.

This example shows the most complete form of combination that the current design can support. The two plot lines involved in the generation of the story contained a good number of shared compatible roles, so the outcome looks quite united. In addition, the scenes from the first plot perfectly alternate with the scenes from the second one. In this case, the postcondition of every scene from the first plot has been compatible with the precondition of the equivalent from the second plot, but this is not necessarily the norm. In many cases, the Plot Weaver will need to pass to the next scene in the first plot until it can insert the scene in the second. Moreover, the number of scenes in the plots to interweave can perfectly be different, and this will entail that the mixed plot line will contain several consecutive scenes from the same plot line.

It is also worth to mention that, in many cases, the preconditions and postconditions basically refer to facts that rarely affect to scenes from different plot lines. This means that there could be many combinations in terms of scene ordering that also would keep the global consistency of the story. The resulting ordering is mainly due to the previously described interweaving procedure, which always takes first a scene from the first plot and tries to find the next compatible scene in the second plot. Table 4 shows several examples of this. For example, the scenes “A peaceful community” and “Frustrated character” are interchangeable without affecting the consistency of the story, as well as the “Outsider destructive actions” and “Pact with evil”. On the other side, the “Temptation” scene will be pointless if it happened before the “Arrival of the outsider”. In this case, the Plot Weaver would have applied the procedure to establish a consistent ordering of the scenes.

Conclusions and future work

The presented adaptation entails a good number of validations in terms of knowledge representation and consistency. The assurance of a consistent merging of two different plot lines, putting together episode by episode from the two plots, is not an easy task. Despite the Plot Weaver checks the proper fitting of the respective precondition and postcondition of the episodes, there can be inconsistencies at a global level.

A significant case that can easily occur in the current model is the reappearance of a character killed in an episode of one plot in later episodes of the other plot, creating a kind of blocking inconsistency—the affected plot could not continue in a consistent way with the merged plot. This is caused by the fact that, despite the match of the corresponding preconditions and postconditions, the current version of the Plot Weaver does not consider the whole plot line, so there can emerge inconsistencies from a global point of view. For example, in one of the two plot lines, a common character can die. It is perfectly possible that this character does not appear in the episode that follows the one in which he / she dies. This circumstance makes the postcondition of the first episode to be consistent with the precondition of the following one. But, the character suddenly appears later, in an episode from the plot line which did not include the death of this character. This situation can be amended by including long-term conditions, that are propagated across the whole plot lines. In further iterations, these checks will be faced to guarantee a fully consistent story. On the other hand, from a positive point of view, this kind of situations could be interesting for developing stories according to an unnatural narrative plan, what could be specifically explored in a future line of research. The next natural step in this adaptation process will be the development of the mechanism that holds this need. The evolution of this model will provide generated lessons that will be helpful for making better decisions, and in marking paths for future investigation. One of this paths can be deepening in the development of a more pervasive plot interweaving, in which a Plot Weaver works with incomplete drafts. In this scenario, the plot generation and the episode generation stages in the different generation stages would directly interchange events between them, during their own activity. This approach, more decentralized in the sense of the events will not be mediated by the Story Director, would bring more creative wealth, but also more complexity to the process.

The Plot Weaver supports the application of different weaving strategies. One of the promising candidates to be considered in future versions is the “Chinese Box” technique (Menéndez 2013), which considers the development of nested plot lines inside a larger one. This adaptation would entail not only a modification of the Plot Weaver strategy, but also the model of generation applied by the Plot Generator.

Another aspect that can be analyzed in the future is the ability of merging more than two plot lines in a single story. Many of the already designed heuristics will still be useful, but probably we would need to design new ones to address the complexities associated to this new requirement and introduce more thorough draft evaluation mechanisms (Gervás and León 2016; Tapscott et al. 2016).

Acknowledgments

This paper has been partially funded by the projects IDiLyCo: Digital Inclusion, Language and Communication, Grant. No. TIN2015-66655-R (MINECO/FEDER) and InVITAR-IA: Infraestructuras para la Visibilización, Inte-

gración y Transferencia de Aplicaciones y Resultados de Inteligencia Artificial, UCM Grant. No. FEI-EU-17-23.

References

- Alber, J., and Heinze, R. 2011. *Unnatural narratives-unnatural narratology*, volume 9. Walter de Gruyter.
- Balló, J., and Pérez, X. 2007. *La semilla inmortal: los argumentos universales en el cine*. Ed. Anagrama.
- Bellifemine, F.; Poggi, A.; and Rimassa, G. 1999. Jade—a fipa-compliant agent framework. In *Proceedings of PAAM*, volume 99, 33. London.
- Bringsjord, S., and Ferrucci, D. 1999. *Artificial intelligence and literary creativity: Inside the mind of brutus, a storytelling machine*. Psychology Press.
- Cervantes, M. 2011. *Don Quixote*. Penguin Random House.
- Cohen, P. R., and Feigenbaum, E. A. 2014. *The handbook of artificial intelligence*, volume 3. Butterworth-Heinemann.
- Concepción, E.; Gervás, P.; and Méndez, G. 2017. A common model for representing stories in automatic storytelling. In *6th International Workshop on Computational Creativity, Concept Invention, and General Intelligence. C3GI 2017*.
- Concepción, E.; Gervás, P.; and Méndez, G. 2018a. Afanasyev: A collaborative architectural model for automatic story generation. In *5th AISB Symposium on Computational Creativity. AISB 2018*.
- Concepción, E.; Gervás, P.; and Méndez, G. 2018b. Ines: A reconstruction of the charade storytelling system using the afanasyev framework. In *Ninth International Conference on Computational Creativity, ICC3 2018*.
- Dehn, N. 1981. Story generation after tale-spin. In *IJCAI*, volume 81, 16–18.
- Fay, M. P. 2014. *Driving story generation with learnable character models*. Ph.D. Dissertation, Massachusetts Institute of Technology.
- Flaubert, G. 1857. *Madame Bovary*. Michel Lévy.
- Gervás, P., and León, C. 2014. The need for multi-aspectual representation of narratives in modelling their creative process. In *5th Workshop on Computational Models of Narrative, OASICS-OpenAccess Series in Informatics*.
- Gervás, P., and León, C. 2016. Integrating purpose and revision into a computational model of literary generation. In *Creativity and Universality in Language*. Springer. 105–121.
- Gervás, P. 2012. Story generator algorithms. In *The Living Handbook of Narratology*. Hamburg University Press.
- Gervás, P. 2018. Storifying observed events: Could i dress this up as a story? In *5th AISB Symposium on Computational Creativity*. University of Liverpool, UK: AISB.
- Glassner, A. 2009. *Interactive storytelling: Techniques for 21st century fiction*. AK Peters/CRC Press.
- Lebowitz, M. 1984. Creating characters in a story-telling universe. *Poetics* 13(3):171–194.
- Li, B., and Riedl, M. O. 2010. An offline planning approach to game plotline adaptation. In *AIIDE*.
- Liu, H., and Singh, P. 2002. Makebelieve: Using common-sense knowledge to generate stories. In Dechter, R., and Sutton, R. S., eds., *AAAI/IAAI*, 957–958.
- Machado, I.; Paiva, A.; and Brna, P. 2001. Real characters in virtual stories. In *International Conference on Virtual Storytelling*, 127–134. Springer.
- Martínez, A. 2011. *Escribir teatro*. Barcelona, Spain: Alba Editorial.
- Meehan, J. R. 1977. Tale-spin, an interactive program that writes stories. In *In Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, 91–98.
- Méndez, G.; Gervás, P.; and León, C. 2014. A model of character affinity for agent-based story generation. In *9th International Conference on Knowledge, Information and Creativity Support Systems*, volume 11.
- Méndez, G.; Gervás, P.; and León, C. 2016. On the use of character affinities for story plot generation. In *Knowledge, Information and Creativity Support Systems*. Springer. 211–225.
- Menéndez, R. 2013. *Cinco golpes de genio*. Barcelona, Spain: Alba Editorial.
- Mott, B.; Lee, S.; and Lester, J. 2006. Probabilistic goal recognition in interactive narrative environments. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, 187.
- Pemberton, L. 1989. A modular approach to story generation. In *Proceedings of the fourth conference on European chapter of the Association for Computational Linguistics*, 217–224. Association for Computational Linguistics.
- Perez y Perez, R. 1999. *MEXICA: A Computer Model of Creativity in Writing*. Ph.D. Dissertation, The University of Sussex.
- Porteous, J.; Charles, F.; and Cavazza, M. 2016. Plan-based narrative generation with coordinated subplots. In *European Conference on Artificial Intelligence (ECAI 2016)*, volume 285, 846–854. IOS Press.
- Prada, R.; Machado, I.; and Paiva, A. 2000. Teatrix: Virtual environment for story creation. In *International Conference on Intelligent Tutoring Systems*, 464–473. Springer.
- Propp, V. 1968. Morphology of the folk tale. 1928.
- Riedl, M. O., and Young, R. M. 2010. Narrative planning: balancing plot and character. *Journal of Artificial Intelligence Research* 39(1):217–268.
- Tapscott, A.; Gomez, J.; León, C.; Smailovic, J.; Znidarsic, M.; and Gervás, P. 2016. Empirical evidence of the limits of automatic assessment of fictional ideation. In *C3GI@ESSLLI*.
- Tarantino, Q. 1994. Pulp fiction.
- Todorov, T. 1975. *The fantastic: A structural approach to a literary genre*. Cornell University Press.
- Vernet, J., ed. 1990. *The Arabian Nights*. Editorial Planeta.
- Winston, P. H. 2016. The genesis story understanding and story telling system a 21st century step toward artificial intelligence. Technical report, Center for Brains, Minds and Machines (CBMM).

Generating a Search Space of Acceptable Narrative Plots

Pablo Gervás

Facultad de Informática
Instituto de Tecnología del Conocimiento
Universidad Complutense de Madrid
pgervas@ucm.es

Abstract

Research on narrative generation needs to consider that a run-of-the-mill would-be novelist is usually much more concerned with getting the form of his output right than with finding new techniques or new materials. Resorting to prior techniques and materials is considered an acceptable practice, and rewrites of classic stories, sequels or series of novels set in an already described world are highly valued. In this light the need to produce outputs that are recognisable as instances of stories seems to have priority over other criteria for a narrative generation solution. The present paper follows accepted engineering practice in choosing this particular challenge as starting design goal for an initial module, to be later extended with solutions for selecting or achieving “good” or “novel” stories at a later stage. The paper proposes a representation of plot that captures both a surface structure in terms of adjacency in a discourse sequence and conceptual connections between elements of the plot that may span across its length. Based on this representation, a solution is proposed for plot generation that produces a broad range of outputs that are acceptable as instances of stories.

Introduction

Among the disciplines related to artistic creation, narrative stands out in its relative tolerance for less-than-novel artefacts – as in rewrites, sequels or series – as long as they are well-crafted products that follow established rules of how they are constructed. This is in high contrast with modern art or contemporary music, where vague resemblance to anything that went before is considered very detrimental. This dichotomy is related to the conception of creativity as a balance between novelty and value.

Even though there is not clear agreement on how to define creativity, most accounts consider that for an output to be creative it must show indications of being new and being valuable. This duality was formalised by Ritchie (Ritchie 2007) as a trade-off between typicality and novelty. To be acceptable as valid output, a particular artifact must satisfy criteria to be recognised as an instance of the target class. In Ritchie’s terms, these criteria define typicality for such artifacts. To avoid confusion with biased uses of the term, we redefine this as *acceptability* of an artifact as an instance of the target class. To be considered creative, artifacts need to

be different from outputs already known (novel). There is a tension between these two characteristics. If an item follows very closely the rules for a typical instance of its kind, it is unlikely to be considered novel. If an artifact innovates radically with respect to prior instances, it runs the risk of not being considered typical, even to the point where it is no longer considered an instance of the target class.

The criteria applied to make these decisions vary across genres. In modern art, for instance, novelty is valued very highly, and typicality of any kind is almost frowned upon. An art piece remotely resembling prior production in any aspect immediately loses points in the appreciation scale being applied tacitly. This includes physical resemblance, choice of material, or techniques employed in its construction. The objection that an artist in search of recognition fears the most is that of not being novel enough. In the realm of narrative, in contrast, although there is still significant pressure on finding new techniques or new materials, there is a major concern that the results be intelligible, and these imposes important concerns on how much novelty can be introduced without compromising the ability to communicate with the reader.¹ It is also true that the reuse of elements from prior stories is considered an acceptable practice (Tedford Jones 2002). This includes reuse of the structure of a previous narrative – rewrites of classic stories –, or the characters – sequels –, or the setting – series of novels set in the same world. A would-be novelist is much more concerned with getting the form of his output right (having learnt how to write in the acceptable fashion) than with minimizing his reference to the classics. In fact, having frequent reference to prior work, or resorting to known successful tropes is often done consciously in search of this impression of having mastered the craft.

From the point of view of research on narrative generation, it is clear that for this particular genre acceptability is valued highly and novelty is assigned less importance. In terms of techniques and procedures to be employed, this implies that the main consideration to apply when selecting or

¹Although the topic is controversial, interested readers can refer to (Lodge 1981) – “if a novel did not bear some resemblance to other novel we should not know how to read it” – or (Anderson 2007) – “novelty (...) threatens to present a major obstacle to how and what these text can communicate to readers.” – for more detailed discussion.

designing a narrative generation solution is to ensure that it will produce outputs that are clearly recognisable as instances of stories. A would-be designer of narrative generators needs to invest a significant portion of his time in developing techniques that will capture the unwritten laws of how to write, master the known tropes, and are aware of classic references in the field. This does not mean that novelty can be ignored. When reusing known structures new ways of instantiating them must be found or new twists added to give them a spark.

This need to strike a delicate balance between reproduction of known elements and innovation has traditionally been resolved by the application of knowledge-based approaches where knowledge is defined in meaningful units that capture the ingredients to be reproduced, and procedures are devised for recombining them into new material in a way that captures the essence of the craft without reproducing existing material literally. Finding the right balance is a significant challenge. If the reference material is represented with small granularity, local coherence of the results is assured but there may be insufficient knowledge to drive an interesting overarching plot. If the granularity is too large, the structure of the results does not depart from that of the reference material enough to suggest novelty.

The present paper reviews existing approaches to story generation focusing on the granularity at which they represent the knowledge that drives plot construction, and discusses their relative merits. Based on evidence arising from this analysis, a new plot representation format is proposed that combines the advantages of low and high granularity solutions, and a construction algorithm designed to guarantee the acceptability of output stories is presented.

Related Work

The work presented in this paper is inspired by and departs from some prior theoretical accounts of plot and some computational approaches to the representation and construction of narrative.

Some Relevant Theoretical Accounts of Plot

The most popular representation schema for plot among the early story generators has been the concept of *character functions* as presented in the morphology of the Russian folk tale (Propp 1968). A character function for Propp is an abstraction over certain actions of the characters that are relevant to the overall plot. Examples of these actions relevant for the plot are: performing a villainous act, starting a fight, winning a fight, departing on a journey or returning from a journey, but also less active choice points in the story such as deciding to take action in view of a villainy, recognising a character that was in disguise, or rewarding someone. For Propp, these character functions, in their abstract versions, were shared across the set of folk tales he analysed. He considered them grouped into a set of spheres, each one of them associated with one of the dramatis personae: the hero, the villain, the victim... Furthermore, Propp postulated an overarching canonical sequence that described the relative order in which the character functions appeared in the plot.

Computational solutions based on Propp's account have tended to borrow some parts of the analysis while forgetting others. The part of Propp's account most often used are his character functions, which correspond to representing the plot with a small granularity.

At the opposite extreme of the spectrum, there have been a number of efforts to represent plot in terms of its overall structure as a whole. These efforts postulated a number of abstractions of the structure of a plot that act as the set of master plots available for building stories. The number of such master plots varies between the single plot structure known as *the hero's journey* (Campbell, Cousineau, and Brown 1990) and Plotto's 1,462 plots (Cook 2011) with intervening values at twenty (Tobias 2012) or seven (Booker 2004). This diversity in values relates to the level of abstraction at which the plots are described. This in itself presents a serious challenge for a researcher hoping to establish an appropriate representation scheme, but it also suggests that a solution articulated at a lower level of granularity, which described complex plots as combinations of smaller units might be better suited to capture the complexity in the data in a more efficient way.

Although many more accounts of plot from a theoretical point of view have been reviewed to inform the present research, it is beyond the scope of this paper to list them all. But there are two specific ones that suggest a middle way to the dilemma discussed so far. Working at different level of granularity in representing his material, Polti describes thirty six dramatic situations (Polti and Ray 1916) that can be used by a playwright to structure his material. On close perusal these situations are not full fledged plots. The examples provided in the book very rarely intend the situation in question as a description of the complete plot, and in describing them the author often refers to particular acts of the plays in question as instances of these situations. Although no definition of a situation is given, they are at a certain point referred to as "actions possible to the theater". It seems they are intended as building blocks for a plot. Some of them correlate reasonably well with certain of Propp's functions (abduction, pursuit) but others seem to operate at a larger degree of granularity than Propp's character functions (crime pursued by vengeance). Another important insight can be obtained from Forster's analysis of plot (Forster 1927). For Forster, plot is distinct from a chronological sequence of events in that the events within a plot must be connected by some kind of causality that drives the sequence and gives it meaning. These two sources suggest that a representation at an intermediate level of granularity, coupled with a procedure for explicitly connecting them to one another with some kind of motivational link, could provide a solid basis for representing plot.

Some Computational Representations of Plot

Existing efforts at representing plot with generative purposes may also be analysed in terms of the granularity at which they consider their knowledge units. In each case, they also consider additional procedures for governing how the individual units are combined into output plots.

There are many story generators that rely on planning so-

lutions (Riedl 2004), based on the assumption that a plan – which draws a connecting path between an initial situation and a goal – has a certain parallelism with a story. In these approaches, the basic unit for representing plot is a planning operator, which involves a representation of an action or event, with associated preconditions and postconditions. Local coherence is achieved by the precondition/postcondition links between actions in a plot, and the overall structure is controlled by the definition of an initial situation and a goal that the plot will conform to. The planning approach to generating plot provides very strong local coherence and weak control over the story arc over a complete plot, so it has been a preferred solution for interactive storytelling, where allowing the user to interfere with the story at will with no obvious loss of coherence is actually a virtue. Attempts have been made to enhance the control over the structure of the plot by the use of *vignettes*, which are defined as “plot fragments that are a priori known to be good” (Riedl and Sugandh 2008; Riedl and León 2008). This approach involves adding an intermediate level of granularity to the representation of plot, one where a set of actions are already packed into a larger fragment that is reused in the construction process.

The Mexica system (Pérez y Pérez 1999) also relies on a combination of *story actions* – atoms of action defined with preconditions and postconditions – with larger structures called *story contexts* that determine how such story actions can be combined. Story contexts are extracted from a set of prior stories that established the knowledge base of the system.

A different family of narrative generators relies on story grammars (Rumelhart 1975) to represent story structure (Lang 1999; Bringsjord and Ferrucci 2000). Rumelhart’s original proposal for story grammars included a syntactic component – which establishes the sequence of plot tokens in the surface form – coupled with a semantic component – which imposes restrictions on conceptual relations that should hold between tokens adjacent in the plot sequence for the story to be meaningful. This combination matches Forster’s concept of plot as requiring conceptual connections across story elements. Computational approaches based on Rumelhart’s story grammars have attempted to implement the semantic component in different ways, such as additional Prolog clauses for the semantics (Lang 1999) or satisfaction of a theory of betrayal (Lang 1999; Bringsjord and Ferrucci 2000).

Propp’s formalism for representing plot in terms of character functions has been employed repeatedly in historic systems. The most faithful rendering of Propp’s approach was developed in (Gervás 2015), where plots were represented as sequences of plot atoms represented as instances of Propp’s character functions, and combined based on preconditions and particular heuristics to capture long term dependencies between them. These long term dependencies were represented in the system as a specific additional knowledge resource.

Many of these approaches to plot generation have been compared in terms of their relative ability to satisfy specific metrics related to quality of the resulting plot structures

(Gervás 2017). The conclusions of that comparison were that different approaches – and the associated representations – focus on particular features that are necessary in a story but not altogether sufficient. The combined set of features so identified was not covered by any of the individual approaches. The solution proposed in this paper attempts to capture a broad combination of the features in a single representation.

More recent work on the generation of fictional stories based on observed facts (Gervás 2018a) – referred to as *storification* – rely on a representation of plot as a sequence of *plot elements*, where each plot element is similar to a character function but explicitly holds additional information to indicate how the roles specific to the plot element (kidnapper, kidnapped) are filled in by roles that are relevant to the plot (villain, victim). In this work, plots are applied only as fixed set of schemas represented in this form, which are matched to observed facts to reach a joint representation that combines fact and fiction into a new story. An extension of this work (Gervás 2018b) first proposed the concept of *axis of interest* as a representational mechanism for articulating the representation of plots. In that approach, axes of interest were only used as representational devices, to allow for the construction – by hand – of a broader set of plot schemas to use as resources in the storification process.

Generating Acceptable Narrative Plots

Given that generation of narrative plots is such a complex task, one possible approach to simplify the engineering of a system might be to first model the ability to create acceptable narrative plots – in the same way as would-be authors first have to master the craft of generating acceptable stories – and later refine the approach to focus on “good” or “novel” stories. By concentrating on the simpler problem of generating acceptable stories (regardless of their novelty) the development task can focus on solving the difficulties involved in achieving acceptability. If a procedure is found to generate a broad range of acceptable stories, the achievement of quality and novelty may be attempted later by the application of metrics to filter the outputs. This would correspond to the *filtration* approach to the construction of generative systems, defined in (Ventura 2016) among the ones with an option for being considered beyond “mere generation”. The present paper focuses on the first stage of such an approach: the generation of a search space of acceptable narrative plots.

The approach followed to achieve this involves composing plot schemas as the interweaving of a number of linear substructures, themselves built up of conceptually interconnected plot atoms.

Representing Plot

The plots considered for the present paper are represented in terms of structured compositions of basic units called *plot atoms*. A plot atom is built along the lines of the plot elements defined in prior approaches to storification (Gervás 2018a; 2018b): a unit similar to a character function which explicitly holds additional information to indicate how the

AXISofINTEREST =	DONOR
AXISofINTEREST PROTAGONIST =	tested
AXISofINTEREST ROLES =	tested tester user gift
<hr/>	
PLOT-SPAN-NAME =	Tested
Tested	characters(tested=X,tester=Y)
Character'sReaction	characters(tested=X,tester=Y)
ProvisionOfAMagicalAgent	characters(tested=X,tester=Y)
	objects(gift=Z)
<hr/>	
PLOT-SPAN-NAME = UseOfAMagicalAgent	
UseOfAMagicalAgent	characters(user=X)
	objects(gift=Z)

Table 1: The Axis of Interest for the DONOR sequence (Propp 1968). Upper case letters indicate free variables.

roles specific to the plot element (kidnapper, kidnapped) are filled in by roles that are relevant to the plot (villain, victim). This refinement allows for interesting articulation between roles specific to a plot atom and more general roles that refer to the overall plot.

The plot atoms in a plot are organised in a complex structure that combines different sequences of plot atoms. This is required to allow for the concepts of plots that relate actions that take place at non-contiguous points in time (villainy early in the story, revenge at the end of it, with other unrelated events happening in between), or plots that combine more than one subplot (each subplot is a different sequence of plot atoms which may be interleaved with the other subplots by breaking their sequence down into smaller subsequences that constitute different scenes of the subplot). This type of complex structure is represented by a recursive data structure: the plot span. A *plot span* represents a span of plot, constituted by a sequence of plot atoms (or smaller spans). The idea is to capture the concept of a number of plot atoms appearing as a structural unit in a plot, but not necessarily occurring contiguously in the discourse for the plot. Plot spans of this type can be used to represent complete plots. When a plot span represents a complete plot, it is called a *plot schema*. Plot spans can also be used to describe intermediate units of plot structure that involve a set of plot atoms related by a long range dependency. These are called *axes of interest*. For example, a plot span representing an *Abduction* as it features in classic stories would include the actual kidnapping (which would happen somewhere towards the start of the story) and the corresponding *Release* (which would happen somewhere towards the end of the story), but these two plot atoms are structurally connected. An axis of interest has a set of narrative roles – those of its constituent plot atoms – that are initially free variables but which can be instantiated to specific constants when the axes of interest is combined into larger structures.

Two examples of axes of interest are shown in Tables 1 and 2. To assist in the process of combining them into more elaborate structures, each axis of interest specifies which character is the protagonist and what the roles relevant to the axis of interest are.

Axes of interest can be combined together, weaving their corresponding subspans with those of other axes of interest, to form plot schemas. A *plot schema* encodes the way in

AXISofINTEREST =	CONFLICT
PROTAGONIST =	attacker
ROLES =	attacker defender winner looser
<hr/>	
PLOT-SPAN-NAME =	Struggle
Struggle	characters(attacker=X,defender=Y)
<hr/>	
PLOT-SPAN-NAME =	Victory
Victory	characters(winner=X,looser=Y)

Table 2: The Axis of Interest for CONFLICT

PLOT-SCHEMA =	OCM-DonFight	
PROTAGONIST =	hero	
<hr/>		
DONOR	Tested	characters(tested= hero ,tester=donor)
		objects(gift=gift)
<hr/>		
CONFLICT	Struggle	characters(attacker= hero ,defender=villain)
DONOR	UseOfAMagicalAgent	characters(user= hero)
		objects(gift=gift)
<hr/>		
CONFLICT	Victory	characters(winner= hero ,looser=villain)

Table 3: Example of plot schema for a basic plot combining axes of interest for DONOR and CONFLICT. The first column shows the interweaving of the axes of interest. Horizontal lines show the boundaries between spans corresponding to different axes of interest. The co-occurrence of constants on both sides of a boundary line in the final column – shown in **bold** – indicates the presence of a plot link at that point between the two axes.

which several axes of interest combine together to form the plot span for an elaborate plot. In a plot schema, the plot atoms from the axes of interest that have been combined appear in an ordered sequence that corresponds to the discourse for the plot schema, but each plot atom is labelled to indicate which axes of interest it corresponds to. Additionally, the plot schema lists for each plot atom how the roles specific to the various axes of interest are instantiated in terms of the set of constants that encode the overall set of narrative roles involved in the plot schema.

An example of plot schema is presented in Table 3, which shows how the DONOR and CONFLICT axes of interest – both abstracted from Propp's account of the Russian folk tale – are interleaved to form a very basic plot where the hero defeats the villain using a magical agent acquired earlier in the story. It also shows how the narrative roles for the plot (*hero*, *villain*, *victim*, *donor*) are mapped to the roles specific to the plot atoms of the constituent axes of interest (*tested*, *tester*, *user* for the DONOR axis of interest and *attacker*, *winner* for the CONFLICT axis of interest). This ensures that the various plot atoms in the plot are instantiated in a manner coherent with the narrative roles that the characters play in the overall plot schema.

Restrictions on Axes of Interest Combination

For a plot schema to be considered a valid narrative plot, the variables in the axes of interest that compose it must be instantiated in a coherent manner. For the example in Table 3, the required restrictions can be expressed by requiring that the character who acts as *tested* and *user* in the DONOR axis be the same one who acts as *attacker* and *winner* in the

DONOR	ProvisionOfAMagicalAgent	tested
CONFLICT	Struggle	attacker
CONFLICT	Struggle	attacker
DONOR	UseOfAMagicalAgent	user
DONOR	UseOfAMagicalAgent	user
CONFLICT	Victory	winner

Table 4: Plot links for the connections between the DONOR and the CONFLICT axes of interest in the context of the plot schema in Table 3. Each plot link is described by two lines in the table. The first column indicates the axes of interest involved, the second column indicates the plot atoms appearing at the boundaries, and the third column indicates the roles in each axis that need to be instantiated to a shared character. A plot link exists by virtue of the pairs of axis-specific roles in the third column being both instantiated with the same character at different sides of a boundary between two adjacent spans.

CONFLICT axis. Note that unless this condition holds the proposed plot schema makes no sense. These restrictions across axes of interest are captured in terms of *plot links* such as the one shown in Table 4.

If the plot schema is created by hand, the plot links can be abstracted from it by identifying shared variables occurring at boundaries of adjacent spans. Plot links extracted in this fashion can then be used to guide recombination of existing axes of interests into new plot schemas according to the construction procedure described below.

A sequence of plot atoms from several axes of interest, interwoven into a single linear sequence, is considered *valid from a narrative point of view* if at any point in the sequence where plot atoms from different axes of interest appear contiguously, there exists a plot link connecting a shared variable.

A plot schema is considered valid if the sequence of plot atoms it encodes is valid.

The simplicity of representing plots in this way allows for the rapid construction of a large number of variations of simple plots by combining a reduced set of axes of interest, while allowing for significant structural complexity in the resulting plots, arising from the interleaving of the axes of interest.

The search space of possible plots obtainable with this representation can be generated by exploring all combinations of the available axes of interest. A combination of two sequences of plot atoms is built by considering all possible interleavings of the plot atoms in them, and pruning any branches of this search where plot atoms from different axes of interest appear contiguously and are not supported by plot links.

Knowledge Engineering Issues

A set of plot atoms and axes of interest built by combining them needs to be crafted by hand. This is a significant knowledge engineering effort, along the lines of others previously described in the literature on narrative generation (Gervás, León, and Méndez 2015). For the sake

Number of axes of interest combined	Number of plots generated
2	144
3	1,051
4	10,301

Table 5: Number of plots generated for combinations of different numbers of axes of interest.

of comparability, the set of basic plot atoms proposed in (Gervás, León, and Méndez 2015) was considered, together with a set of 19 axes of interest corresponding to the subsequences of Propp’s canonical sequence (Propp 1968), and the elementary encoding of instances of Booker’s seven basic plots (Booker 2004) proposed in (Gervás, León, and Méndez 2015).

The proposed representation has the advantage that the set of plot links can be mined from a set of instances of plot schemas built using this vocabulary of basic elements. For the purposes of this paper, an initial set of 34 plot schemas was constructed to act as seed. A set of plot links was created by means of a bootstrapping procedure:

- the set of seed plots was parsed to obtain an initial set of plot links
- the set of plot links so obtained was used to drive a construction procedure that generated a set of new plot schemas by exploring pairwise combinations of the existing axes of interest
- the set of output plot schemas was manually revised for correctness and expanded by the construction of further combinations analogous to the ones in the output
- the revised and expanded set of plot schemas was parsed to obtain further plot links

This procedure resulted in a set of 221 plot schemas, which gave rise to a set of 423 plot links between the 34 plot atoms over the 19 axes of interest.

Testing Generative Capacity

The resulting set of resources can then be used to generate combinations of larger numbers of axes of interest. Overall numbers of plots generated for different values of the number of axes of interest considered are given in Table 5.

The amount of outputs generated makes it impractical to carry out an exhaustive quality check. Random sampling was carried out by generating a number at random within the range of numbered outputs and checking the corresponding plot. The examples presented below have been chosen in this fashion. Overall they seem to be acceptable as possible instances of plots, and some of them are actually reasonable in the sense that one can follow a certain logic in the way that the plot atoms follow one another.

The system generates conceptual descriptions of plot schemas much along the lines of the examples of knowledge resources presented earlier. For ease of reading, a simple template-based text realizer has been developed that converts such conceptual representations into readable text. The

RIVALRY	Rivalry	Hero develops rivalry with shadow.
CROSSDRESSING	CrossDressing	Hero dresses up as a member of the opposite sex.
PURSUIT	Pursuit	Hero is pursued by villain.
RIVALRY	Cooperation	Hero cooperates with shadow.
RIVALRY	RivalReconciliation	Hero ends rivalry with shadow.
PURSUIT	RescueFromPursuit	Hero avoids pursuit.
CROSSDRESSING	Recognition	Hero is recognised.

Table 6: Example output for combination of 3 axes of interest: RIVALRY, PURSUIT and CROSSDRESSING.

RAGS2RICHES	Poverty	Hero suffers poverty.
RAGS2RICHES	Aspiration	Hero has aspiration.
RELENTINGGUARDIAN	CoupleWantsToMarry	Hero wants to marry love-interest.
RELENTINGGUARDIAN	UnrelentingGuardian	Obstacle objects to proposed union of hero with love-interest.
TASK	DifficultTask	Hero is set a difficult task by obstacle .
TASK	Solution	Hero solves the task.
RAGS2RICHES	Transformation	Hero is transformed.
RELENTINGGUARDIAN	RelentingGuardian	Hero convinces obstacle
RELENTINGGUARDIAN	Wedding	Hero marries love-interest.
RAGS2RICHES	Reward	Hero is rewarded.

Table 7: Example output for combination of 3 axes of interest: RAGS2RICHES, TASK and RELENTINGGUARDIAN.

quality and elaboration of the resulting texts has deliberately been kept low, to avoid confusion between any merits arising from the conceptual structure of the narrative plots generated and any beauty that may arise from the texts generated to render these structures.

Results for different combinations of 3 axes of interest are shown in Tables 6, 7 and 8. In each case the first two columns show the axes of interest and the plot atoms involved respectively, and the third column shows the output story rendered as text. Boundaries between plot spans corresponding to different axes of interest are indicated by horizontal lines. The characters instantiating the plot links across the boundaries are shown in **bold**.

Tables 9 and 10 show examples for different combinations of 4 axes of interest.

Because they have been obtained via random sampling, the examples presented here should not be considered as selected outputs, but rather as samples out of a very large search space that can be generated by the proposed solution. Because of the high number of potential plot links available, the current procedure generates more than one possible interweaving for a given choice of axes of interest, depending on the choices made over the available plot links. Not all of them are equally fortuitous in terms of the story they represent. For instance, the example in Table 9 might have been a better story if the recognition of the validation received had occurred not immediately after it had been granted, but at a later point in the story, maybe after the hero had cooperated

SHIFTINGLOVE	BoyMeetsGirl	Hero meets and starts relationship with love-interest.
SHIFTINGLOVE	LoveShift	Hero loses the attention of love-interest.
CONFLICT	Struggle	Hero fights with villain.
CONFLICT	Victory	Hero achieves victory over villain.
REPENTANCE	Transformation	Hero is transformed.
REPENTANCE	Repentance	Hero repents.
SHIFTINGLOVE	Reconciliation	Hero makes up with love-interest.
REPENTANCE	RepentanceRewarded	Hero sees repentance rewarded.

Table 8: Example output for combination of 3 axes of interest: SHIFTINGLOVE, CONFLICT and REPENTANCE.

RAGS2RICHES	Poverty	Hero suffers poverty.
RAGS2RICHES	Aspiration	Hero has aspiration.
RIVALRY	Rivalry	Hero develops rivalry with shadow.
CROSSDRESSING	CrossDressing	Hero dresses up as a member of the opposite sex.
VALIDATOR	Tested	Hero is tested by validator.
VALIDATOR	Character'sReaction	Hero reacts to the test by validator.
VALIDATOR	Validation	Hero is validated by validator.
VALIDATOR	ValidationRecognised	Hero sees validation recognised.
RAGS2RICHES	Transformation	Hero is transformed.
RIVALRY	Cooperation	Hero cooperates with shadow.
CROSSDRESSING	Recognition	Hero is recognised.
RIVALRY	RivalReconciliation	Hero ends rivalry with shadow.
RAGS2RICHES	Reward	Hero is rewarded.

Table 9: Example output for combination of 4 axes of interest: RAGS2RICHES, RIVALRY, CROSSDRESSING and VALIDATOR.

SHIFTINGLOVE	BoyMeetsGirl	Hero meets and starts relationship with love-interest.
SHIFTINGLOVE	LoveShift	Hero loses the attention of love-interest .
JOURNEY	Departure	Love-interest departs.
TASK	DifficultTask	Dispatcher sets a difficult task to love-interest .
PURSUIT	Pursuit	Love-interest is pursued by villain.
PURSUIT	RescueFromPursuit	Love-interest avoids pursuit.
TASK	Solution	Love-interest solves the task.
JOURNEY	Return	Love-interest returns.
SHIFTINGLOVE	Reconciliation	Hero makes up with love-interest .

Table 10: Example output for combination of 4 axes of interest: TASK, PURSUIT, SHIFTINGLOVE and JOURNEY.

with the shadow. Such a combination may have occurred in the set of outputs, but checking them by hands to find it is a serious task.

Overall, the sampling carried out over the output suggests that there are no seriously flawed results. Some of the stories are more interesting than others, but this is to be expected in an exhaustive enumeration of the search space afforded by the chosen representation. As it stands, the proposed procedure constitutes a valuable initial module over which processes of filtering or selection may be developed by the introduction of appropriate metrics designed to capture particular concepts of story quality.

Discussion

The proposed system is discussed in terms of some of its shortcomings and in terms of its relation to prior work.

Aspects in Need of Improvement

The experiments carried out in support of the writing of this paper have uncovered a number of issues that may need attention.

The current approach to the representation of connections between plot atoms within a plot schema is based on the identification of shared variables representing characters that participate in each of the plot atoms with a different role. This is adequate for capturing a significant number of connections, but some connections very relevant to traditional stories are beyond the reach of the formalism. For instance, many classic plots are initiated by a villainy that results in the hero being faced with the call to action (Propp 1968; Campbell, Cousineau, and Brown 1990). The current version of the formalism cannot capture this kind of connection, because the hero is not explicitly mentioned in the villainy and the villain is not explicitly present in the call to action.

So no plot link can be established between them. This occurs in a number of further instances of plot where there is a conceptual connection between adjoining plot atoms that is relevant to the structure of the story but does not rely on co-occurrence of any character across the pair. This is one of the reasons why the number of generated plots falls below the expected number of possible combinations of the resources invested – the set of plot links mined from 221 seed plots produces only 144 constructed plots, as described above. The underlying formalism for representing plots may be revised in the future to address this limitation.

Another important shortcoming of the proposed formalism in its current version is that, due to the limitations of the initial implementation of the procedure for instantiating variables when weaving plot spans together, it does not allow for an axis of interest to be used more than once in a given plot. As a result, the system cannot represent a number of classic stories, such as action tales involving more than one fight, fairy tales where the hero attempts a task three times before succeeding, or comedy plots with multiple romantic couples switching affinities between them. This is a significant handicap that needs to be overcome in the future.

Relation to Prior Work

The proposed representation for plot shows significant parallelism with some of the approaches reviewed in the section on *Computational Representations of Plot*.

The decision to enrich the representation of plot atoms with explicit indication of the characters that take part in them, expressed in terms of the roles these characters play in the overall narrative structure of the plot, operationalises the concept of *sphere of action of the dramatis personae* as described by Propp.

It is interesting to see that Propp's overall schema for the representation for plot already includes three levels of representation of plot that match closely those that are proposed in the present paper: one of atoms to be recombined, one of relations between the atoms in terms of characters that play the fundamental roles in them, and one of relative ordering within the general plot arc.

The need for the additional mechanism of plot links to capture a conceptual structure across the plot atoms that is different from and goes beyond their adjacency relations within the sequence of the plot is inspired by Forster's insight that plot incorporates additional levels of connection beyond simple chronological sequence (Forster 1927).

The abstraction of a unit for the representation of plot that is intermediate between a full plot and the kind of plot atom illustrated by Propp's character functions – or plan operators understood as story actions – was already present in Polti's *dramatic situations* (Polti and Ray 1916). Because they sometimes refer to ingredients of the plot structure that span across its length – such as *crime pursued by vengeance* – they have close similarity with the proposed concept of axis of interest.

The use of the concept of an axis of interest to tie together a number of plot atoms into a construction unit larger than an atom but smaller than a full plot has similarity to that of *vignettes* as proposed in (Riedl and Sugandh 2008; Riedl

and León 2008). However, vignettes tend to correspond to short sequences of consecutive actions that fill a single gap at a particular point of time in the plot, whereas an axis of interest will usually span two different moments in time that are relevant for a plot, in a way that allows the encoding of long range dependencies between separate moments of the plot.

By their construction and the way in which they are combined to construct plot schemas, axes of interest are designed to capture conceptual dependencies between plot atoms that are conceptually connected – like an imprisonment and the release of the prisoner – but occur at places in the story distant from one another. This is a significant advantage in that it allows for the construction of structurally complex stories spanned by conceptual links between distant elements.

This use of axes of interest and plot links as extra layers of meaning over the order in which the plot atoms occur in the discourse is related to the need identified in (Rumelhart 1975) to represent the structure of stories at both a more superficial syntactic level and a deeper semantic level. Some of the shortcomings of the solution proposed in this paper arise from the fact that the current representation of conceptual connections in terms of plot links is still too close to the syntactic level represented by explicit mention of a character in a plot atom.

The proposed bootstrapping solution for exploiting the generator itself to construct plot schemas that are then adapted to provide further sources for mining knowledge for the system follows an existing line of work on reducing the bottleneck of knowledge acquisition. Prior solutions to the task of engineering the knowledge resources for story generators had been proposed based on mining crowd-sourced plot graphs (Li et al. 2013) or applying Qualitative Knowledge Engineering methodologies (O'Neill 2013). The procedure for extracting the knowledge resource for driving connections between plot atoms – in the case of this paper, plot links – by parsing instances of stories as represented within the system – here, plot schemas – is borrowed from the way the Mexica system parses prior stories to build its story contexts.

Conclusions

The system described in this paper presents significant advantages in terms of how it captures the conceptual complexity of plot, how it can be used to construct a useful number of knowledge resources required for operation, and how it allows the construction of relatively large number of acceptable plots.

In its current version the system is not intended as a fully creative plot generation system, but rather as a prototype for the initial stage of a developing system. This initial stage would address the task of generating a broad range of acceptable stories instead of aiming for a small set of stories of high quality. In a way, it is intended to model the craft of putting together something that can be recognised as a story, not necessarily a good one.

Even within this scope, the work reported in this paper indicates that some engineering challenges remain unresolved. A number of shortcomings have been identified and further

work on the system will hope to address them to improve both the range and the acceptability of system outputs.

The vocabulary of basic resources – both in terms of plot atoms and in terms of axes of interest – may be extended to increase the expressive power of the proposed representation.

Once those planned improvements have been carried out, long term future work may consider the development of additional modules designed to identify parameters that relate to the perceived quality of stories. If such modules become available, more elaborate procedures may be designed that start to consider the generation a smaller number of stories of better quality, or that focus on generating stories that are significantly different from the stories already known to the system. Consideration of the creativity of the system as a story generator would need to wait upon the development of these additional modules for quality and novelty metrics, and the design of this specialised story generation system.

Acknowledgments

This paper has been partially funded by the project IDiLyCo: Digital Inclusion, Language and Communication, Grant. No. TIN2015-66655-R (MINECO/FEDER) and the FEI-EU-17-23 project InViTAR-IA: Infraestructuras para la Visibilización, Integración y Transferencia de Aplicaciones y Resultados de Inteligencia Artificial.

References

- Anderson, E. H. 2007. Novelty in Novels: a Look at What's New in Aphra Behn's 'Oroonoko'. *Studies in the Novel* 39(1):1–16.
- Booker, C. 2004. *The Seven Basic Plots: Why We Tell Stories*. The Seven Basic Plots: Why We Tell Stories. Continuum.
- Bringsjord, S., and Ferrucci, D. A. 2000. Artificial intelligence and literary creativity: Inside the mind of brutus, a storytelling machine. *Computational Linguistics* 26(4).
- Campbell, J.; Cousineau, P.; and Brown, S. 1990. *The Hero's Journey: Joseph Campbell on His Life and Work*. Collected works of Joseph Campbell. New World Library.
- Cook, W. 2011. *Plotto: The Master Book of All Plots*. Tin House Books.
- Forster, E. M. 1927. *Aspects of the novel*. New York: Harcourt.
- Gervás, P.; León, C.; and Méndez, G. 2015. Schemas for Narrative Generation Mined from Existing Descriptions of Plot. In Finlayson, M. A.; Miller, B.; Lieto, A.; and Ronfard, R., eds., *6th Workshop on Computational Models of Narrative (CMN 2015)*, volume 45 of *OpenAccess Series in Informatics (OASISs)*, 54–71. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
- Gervás, P. 2015. Computational Drafting of Plot Structures for Russian Folk Tales. *Cognitive Computation*.
- Gervás, P. 2017. Comparative evaluation of elementary plot generation procedures. In *6th International Workshop on Computational Creativity, Concept Invention, and General Intelligence*.
- Gervás, P. 2018a. Storifying observed events: Could i dress this up as a story? In *5th AISB Symposium on Computational Creativity*. University of Liverpool, UK: AISB.
- Gervás, P. 2018b. Targeted storyfying: Creating stories about particular events. In *Ninth International Conference on Computational Creativity, ICCO 2018*. Salamanca, Spain: Association of Computational Creativity.
- Lang, R. R. 1999. A declarative model for simple narratives. In *Proceedings of the AAAI Fall Symposium on Narrative Intelligence*, 134–141. AAAI Press.
- Li, B.; Lee-Urban, S.; Johnston, G.; and Riedl, M. O. 2013. Story generation with crowdsourced plot graphs. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence, AAAI '13*.
- Lodge, D. 1981. *Working with Structuralism: Essays and Reviews on Nineteenth and Twentieth Century Literature*. Routledge & K. Paul.
- O'Neill, B. 2013. *A Computational Model of Suspense for the Augmentation of Intelligent Story Generation*. Ph.D. Dissertation, Georgia Institute of Technology, Atlanta, Georgia.
- Pérez y Pérez, R. 1999. *MEXICA: A Computer Model of Creativity in Writing*. Ph.D. Dissertation, The University of Sussex.
- Polti, G., and Ray, L. 1916. *The Thirty-six Dramatic Situations*. Editor Company.
- Propp, V. 1968. *Morphology of the Folktale*. Austin: University of Texas Press.
- Riedl, M., and León, C. 2008. Toward vignette-based story generation for drama management systems. In *Workshop on Integrating Technologies for Interactive Stories - 2nd International Conference on Intelligent Technologies for Interactive Entertainment*.
- Riedl, M. O., and Sugandh, N. 2008. Story planning with vignettes: Toward overcoming the content production bottleneck. In *Interactive Storytelling*, volume 5334 of *Lecture Notes in Computer Science*, 168–179. Springer.
- Riedl, M. 2004. *Narrative Planning: Balancing Plot and Character*. Ph.D. Dissertation, Department of Computer Science, North Carolina State University.
- Ritchie, G. 2007. Some empirical criteria for attributing creativity to a computer program. *Minds and Machines* 17:76–99.
- Rumelhart, D. E. 1975. Notes on a schema for stories. *Representation and Understanding: Studies in Cognitive Science* 211–236.
- Tedford Jones, J. 2002. Depending on memory: Intertextuality in popular fiction. *The Journal of American Culture* 25:81–84.
- Tobias, R. 2012. *20 Master Plots: And How to Build Them*. F+W Media.
- Ventura, D. 2016. Mere generation: Essential barometer or dated concept? In *Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2016)*. Paris, France: Sony CSL.

Field Work in Computational Creativity

Margareta Ackerman^{@*} and Rafael Pérez y Pérez^{+*}

[@]Computer Engineering Department, Santa Clara University. mackerman@scu.edu.

⁺Depto. de Tecnologías de la Información, Universidad Autónoma Metropolitana. rperez@correo.cua.uam.mx

^{*}Both authors contributed equally to this work. Alphabetical ordering is used to signal equal contribution.

Abstract

Numerous scientific disciplines, e.g. social sciences, benefit from field work. What does field work look like in Computational Creativity and what are its potential benefits to research? We refer to the effort of actively making a system or its artifacts widely accessible outside the academia world, and as such getting feedback, as ‘field work’. In this paper, we reflect on our experiences taking our systems, Alysia and MEXICA, out into the wild terrain by making them broadly available. In the case of Alysia, the system itself was made accessible; MEXICA’s artifacts (stories) were shared through a traditionally published book for a broad readership. We consider the utility of field work for these vastly different systems on the CC continuum (Pérez y Pérez 2018), and discuss potential benefits to other research in the area. Finally, we discuss the necessity of developing methodology to enable rigorous registration of knowledge arising from field work in Computational Creativity.

Introduction

An inherently interdisciplinary field, Computational Creativity (CC) draws inspiration from classically creative domains, such as art, literature, and music. Domain experts and (non-expert) broad audiences alike can interface with CC research by providing feedback in the evaluation of autonomous and co-creative systems (Jordanous 2012). Experts can ascertain whether an artifact measures up to higher standards in their creative domain, while feedback from a large broad audience can further help determine the value of an artifact or identify whether a co-creative system is successful in supporting human creativity. Attaining broad audience and expert feedback requires stepping outside of the computational creativity community, often outside the computing discipline all together. We refer to the effort of actively making a system or its artifacts widely accessible, and as such getting this type of feedback, as ‘field work’.

Unfortunately, as in other disciplines, field work in computational creativity is complex and time-consuming. Further, it may actually conflict with goals put forth by tenure and promotion committees, which

often promote a narrow understanding of scientific contributions. Proof-of-concept systems with minimal user interfaces (UI) more than suffice for achieving primary scientific goals, demotivating researchers from devoting time to making their co-creative systems ready for broad user feedback. Creators of autonomous CC systems aren’t directly incentivized to find ways to share the artifacts of their systems with broader audiences.

While many CC systems stay within the academic realm, quite a few researchers have already shared their work with broader audiences (Twitter bots (Veale 2015), DARCI (Norton, Heath, and Ventura 2015), The Painting Fool (Colton 2012), Impro-visor (Keller and Morrison 2007), etc.)

In the absence of formal methodology for CC field work, it is not surprising that work in this direction previously focused on reporting the experience rather than reflecting on the essence of sharing CC systems and artifacts with broad audiences. For instance, one of the largest efforts in the broad exposure of CC has been the musical *Beyond the Fence*, where a multitude of CC systems were used to aid in the creation of a staged musical production. In the 2016 paper on this monumental event (Colton et al. 2016), the authors explain that “This paper acts primarily as a record of the project which led to the *Beyond the Fence* musical and *Computer Says Show* documentaries.” Similarly, (Colton and Ventura 2014) share that the focus of a CC festival that they organized was to “expose audiences to the main ideas of Computational Creativity within a culturally relevant setting, rather than to study audience experiences. Hence, we did not undertake experiments to gauge reactions to the ideas, systems and outputs presented.”

The aim of the current paper is to reflect on the experience of making CC systems and artifacts broadly available, and begin to pave the way towards a methodology for field work in computational creativity. To this end, we share insights resulting from our experience actively sharing our CC systems and artifacts with the outside world. We hope that our analysis may help other researchers decide whether broad exposure is appropriate for their research, encourage others who engage in field work to share their unique insights, and

ultimately lead to a methodical approach to field work in computational creativity.

In this paper, we employ two different systems as frameworks: Alysia (Ackerman and Loker 2017) and MEXICA (Pérez y Pérez and Sharples 2001). Alysia is a co-creative songwriting system. The system originally enabled the creation of vocal melodies for user-provided lyrics. After deciding to increase access to the system, several cycles of extensive user feedback led to radical improvements, including the integration of a co-creative process for lyrics and in-app voices. Alysia was launched on the App Store in January 2019, allowing anyone, regardless of their musical expertise or training, to easily create original songs.

MEXICA is an agent that produces narratives about the old inhabitants of what today is México City. The MEXICA project aims to contribute to the understanding of the creative process. For many years, MEXICA has lived “isolated” inside a laboratory. In December 2017, for the first time, the agent’s stories reached a much broader audience, most of whom did not have computer science or cognitive science backgrounds. This was completely new territory for both MEXICA and its author, Rafael Pérez y Pérez.

This paper attempts to reflect on the broad exposure on Alysia, MEXICA and their designers. Based on the CC continuum (Pérez y Pérez 2018), Alysia is focused on supporting the creative process of human beings while MEXICA attempts to contribute to the understanding of the creative process. We hope to illustrate that, although the fundamental intentions of each of these systems are different, both benefit from exposure to (potentially) massive audiences.

Furthermore, the study of such agents, within the framework we are proposing, allows contrasting their main characteristics: because Alysia is a co-creative system, the audience interacts with both the system itself and its product; while in the case of MEXICA, the audience interacts with an artefact that has gone through a human production process: a book. We hope that this joint exploration will give the reader a broader perspective than considering the systems separately.

Alysia Field Work

Alysia is a co-creative system, made with the aim of helping anyone create original songs. Much like EMI (Cope and Mayer 1996), which was created to help David Cope get out of writer’s block, Alysia was originally made to support my (Margareta’s) desire to write original songs. The first version of Alysia (Ackerman and Loker 2017), which took three months to create, allowed me to write original songs for the first time (notably, after several years of failed attempts at doing so using traditional methods). I was deeply inspired by the first-hand experience of making co-creative CC systems that successfully addressed a challenge that I have been facing for years.

At the time, the system consisted of a co-creative process for making original melodies for user-provided

lyrics. The integration of musical generation with natural language processing was a significant research challenge.

The project quickly became central to my research program, and, unexpectedly got the attention of the media even before its original publication, when it was put on Arxiv. After New Scientist¹, NBC News², and others released articles featuring Alysia, users began to contact us asking to interact with the system. At the time, Alysia was a young system that lacked a user interface and was difficult to install, and as such could not be shared with anyone beyond one-on-one demos.

Driven by the co-creative goals of the research, I wondered how far the interactive aims of Alysia could go. As Alysia was central in helping me achieve my musical goals, I was inspired to explore the bounds of its co-creative potential by sharing it with others. Why not share it with everyone by putting it online? Since all of my work until that point was done in the academic context, it took about two years before I finally decided to take the steps to make Alysia accessible.

I expected that putting Alysia online with a minimal user-interface will give the freedom of self-expression through songwriting to the masses. This did not turn out as expected, which my team and I quickly learned through large volumes of user feedback. It is worth noting that as a publicly available system, user feedback was significantly more extensive, varied, and direct than the feedback we had gotten through more a controlled user survey we did in the academic context³.

We heard from a wide range of users of different ages, musical expertise, and stylistic preferences. Feedback came in a variety of ways. Perhaps most useful was unsolicited feedback of users sharing their experience (positive and negative), and asking for new features. Our team also conducted extensive one-on-one user studies and in-depth surveys.

It quickly became apparent that, even though it was sufficient to let me create original songs, at the time, Alysia was not yet able to provide the same support to everyone else who wished to express themselves musically. It only solved one part: Creating original melodies for lyrics - and even that part needed work. Exposing Alysia to a broad audience quickly revealed its shortcomings, and showed me how far they were from our goals.

¹<https://www.newscientist.com/article/mg23231043-500-machine-learning-lets-computer-create-melodies-to-fit-any-lyrics/>

²<https://www.nbcnews.com/mach/technology/machine-made-melodies-spotlight-artistic-partnership-between-ai-humans-n698486>

³An unpublished manuscript on the system included a user study that compared ALYSIA’s rankings of vocal melodies to how humans would rank the same melodies. While helpful, the insights resulting from that study was limited in scope, particularly when compared with the wealth of diverse feedback received when taking Alysia to broader audiences.



Figure 1: Alysia demo in a middle school. Line up of students waiting to try Alysia first hand. One of many facets of evaluating Alysia via broad audiences pre-launch. Student faces blurred to protect privacy.

Users asked us to assist with lyrics creation, many did not have the production skills to use Digital Audio Workstations to flesh out Alysia’s melodies, and many more struggled with singing. We got a lot of feedback on the melodies themselves: An expert musician pointed out that the melodies consist of too many large intervals, while users wanted melodies that are less varied in both pitch and rhythm (notably, this finding directly conflicted with the user study that we had conducted in the academic context, where ALYSIA’s melodies appeared too monotone based on subjects’ preference for more varied tunes in their ranking). If we were to help people create songs, we had to do a lot more work. This gave our team a huge push forward, and we completed what was originally supposed to be a five year research plan (and more) in less than a year.

We radically improved the melody model, built a co-creative lyrics generator, and integrated in-app voices. We also put forth a new process for song creation that integrated human-made background tracks on top of which the lyrics, melodies, and vocals were created. This led to an end-to-end system that finally achieved the original goal: In extensive user studies, we observed people with no musical experience easily create songs for the very first time.

Thousands of songs have been created with Alysia before it launched on the app store on January 17, 2019. This put the research to an even more rigorous test, by providing large amounts of data that can be used for evaluation. Explicit user feedback is inherently limited and at times misleading. For instance, the most requested feature on an early Alysia beta was the ability to change musical keys. We rushed to add it, only to discover that hardly anyone ever used it.

Implicit feedback, enabled through user data, pro-

vides an exceptionally direct form of evaluation for co-creative systems. Logs reveal how useful users find the co-creative process, letting us see how often they rely on Alysia’s generations, how much they modify them, and how often the users input their own melodies and lyrics. Overall use and retention can be used to gauge the utility of the system.

It is worth pointing out that making Alysia widely accessible required much time and effort spent on UI, marketing, and other activities that are not traditionally integral to research. The effort required to make a co-creative system public is neither feasible nor appropriate for all co-creative systems.

For the Alysia project, making the system accessible to a broad audience led to extensive and rigorous evaluation and radically accelerated research. The impact of unabashedly direct, continuous user feedback cannot be overstated. I believe that Alysia had to be placed in the wild-wild west of the broad consumer market in order to reach its potential. It is possible that other co-creative systems may find similar benefits from wide exposure. Further, it may be worth considering how the CC community may be able to facilitate the exposure of our systems and their artifacts.

MEXICA field work

In December 2017, the book “MEXICA 20 years – 20 experiences” (Pérez y Pérez 2017) started to circulate. The volume includes 20 narratives, each in Spanish and English, generated by a computer agent. Its goal is to offer a different reading experience to the general audience; so, the book does not have technical or scientific intentions.

Previously, MEXICA’s stories have been published in scientific journals (Pérez y Pérez and Sharples 2004; Pérez y Pérez and Sharples 2001; Pérez y Pérez 2015a), conferences (Pérez y Pérez 2014), book chapters (Pérez y Pérez 2015c; 2015b), web pages⁴, specialized talks and so on. However, no such events produced the attention that the book has generated. Why? It is true that the system was improved before generating the tales for the book; for instance, it now generates stories in Spanish and English, the predefined texts are richer (e.g. now it is possible to use pronouns), the analysis of the coherence during the generation process is more robust, the evaluation process is more elaborated, and so on. However, the essence of the system, and therefore the soul of the stories it produces, are essentially much the same. So, why is the general audience giving the system more attention than ever before?

Having a physical book has been a key factor. A book is a familiar artefact that many feel comfortable with. Manuscripts have been amongst us for centuries and so people do not feel threatened by them. In this case, the originality of the cover and the quality of the printed volume has also helped. I also claim that the stories generated by the system are interesting enough to grab

⁴<http://www.rafaelperezyperez.com>

the attention of some readers. Publishing a book usually brings prestige to human authors, sometimes even fame. Thus, when a creative agent becomes the author of a book, there is a good chance that some people will notice it.

Some research domains (e.g. literature) heavily relies on books. Thus, “MEXICA 20 years – 20 stories” works as a bridge that allows me to interact with colleagues from fine arts and literature programs in ways that I did not anticipate. To illustrate my point I would like to introduce Andy Fitch, who is a writer, an editor, teaches in the University of Wyoming’s MFA program, and directs the MA program in literature. He interviewed me for the blog of LA review of books. Thus, the book became a piece that both a computational creativity researcher and a writer felt comfortable discussing. In the following I will show some of the questions that Andy asked me that made me reflect about MEXICA and CC in a different way.

Consideration 1. Social and cultural aspects

Creative computing’s emphasis on cross-cultural inputs also stands out. Harrell’s preface describes this field asking (singing, actually): Must computers always express the voice of the colonizer — could a computer instead express the voices of sovereign indigenous peoples, the oppressed, and the otherwise underrepresented? Could you place these particular questions in a broader cultural context in which we see, for instance, AI processes often absorbing racist biases circulating in U.S. culture, and then further entrenching and institutionalizing those biases? Does MEXICA work against such trends?(Fitch 2018)⁵

It is hard to find people in the CC community interested in studying and analyzing the cultural aspects of CC. A notable exception is the workshop in Computational Creativity and Social Justice, organized by Gillian Smith, Dan Brown and Anne Sullivan, that took place during ICC17. The organizers wrote a report that is available online ⁶. In this document they pointed out, among other things, the necessity of respecting audiences and cultures:

Whose voices are represented in our current technologies, and how does this influence the design of CC technologies? How does the (Irani et al. 2010) notion of postcolonial computing relate to CC? How can we infuse our work with respect for the cultural roots of creativity?....

I believe these are very important questions that seem to be in harmony with Andy Fitch (and Fox Harrel)’s concerns about culture. However, unfortunately, during

⁵<http://blog.lareviewofbooks.org/interviews/computational-cognitive-social-talking-rafael-perez-y-perez/>

⁶<https://cs.uwaterloo.ca/conferences/ccsjw2017/CCSJW17WorkshopReport.pdf>

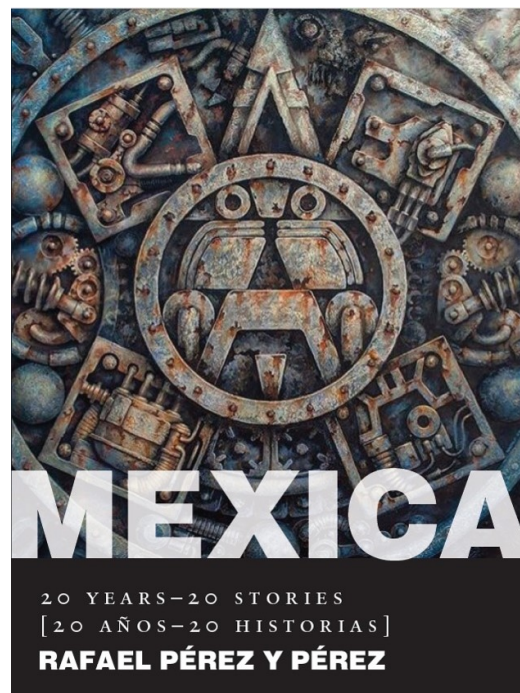


Figure 2: The cover of the book “MEXICA: 20 years, 20 stories”

ICCC18 there were no submissions to a similar workshop and, as a result, it was cancelled. In my experience, CC researchers and students rarely discuss such topics. By contrast, colleagues from the humanities and social sciences seem to believe that creative agents, like MEXICA, might contribute to the debate of these themes. As an example, I was invited to talk about the book and the MEXICA program in: *Crossborders: the aesthetic of immigration* organized by colleagues of the Department of English at the University of Colorado, Boulder, and Counterpath Press in Denver. The organizers describe the event as

a collaborative transnational project that interrogates the cultural and artistic questions that develop from LatinX migration. CrossBorder includes leading scholars, artists, and writers from both sides of the Southern border who are creating work that directly deals with migration, both literal and cultural, of LatinX populations.⁷

CU Boulder Today advertised the meeting as follows: “The free public event series is meant to foster cross cultural dialogues about human migration from Latin America. It will offer attendees a unique combination of academic research and artistic expression” ⁸

⁷<http://counterpathpress.org/crossborders-the-aesthetics-of-migration-at-counterpath-and-cu-boulder-november-9-and-10-2018>

⁸<https://www.colorado.edu/today/2018/11/06/new->

MEXICA was part of the “artistic expression.” Why was I invited? I do not have any idea. Nevertheless, the organizers seemed to be happy with my participation. As a result of all these experiences, I started to wonder how the CC community can contribute to the analyses and study of relevant social concerns such as those mentioned above. I believe that, as community, this is an issue we need to discuss.

Consideration 2. Fresh perspectives on my research

MEXICA the computer system’s calculations might all take place within the confines of a single story. But most readers of MEXICA the book probably absorb much more than one story per sitting... To what extent does the machinery of codex-book construction (with any story always placed alongside other stories, interacting with each other in ways MEXICA does not control, so that encountering the phrase “the princess” in one context might have quite different meanings for readers depending on where else they have encountered that phrase in other MEXICA stories)... seem crucial to the MEXICA project, or seem incidental — just random parts of present-day book circulation? (Fitch 2018)

Andy’s comments made me ponder a novel perspective of the whole project. Because MEXICA’s outputs are single stories, I have always pictured the book as a collection of unrelated narratives. But readers, at least Andy Fitch, seem to perceive the book’s twenty stories as forming a unity. Thus, the idea of improving MEXICA in a way that it can represent the concept of “a collections of related stories”, where characters and actions in one tale are somehow related to those in a different story, seems intriguing. This challenge requires figuring out how to build novel knowledge-structures capable of representing more abstract concepts and how to establish relations between diverse stories. This new perspective relates to Andy’s next question:

You mentioned MEXICA’s minimalist style. I actually thought of minimalist music, with its minimal-event horizons, where you might hear the same note many times, until a very slight change occurs, and this subtle shift suddenly feels like a big deal. By comparison, if MEXICA’s princess has a bad mood for four straight stories, and then in the fifth story feels more ambivalent — even just that muddled mood can register as a significant tonal shift. And those types of structured variations can make MEXICA stories feel quite similar and quite distinct at the same time. (Fitch 2018)

Andy’s thoughts suggested to me that it is not enough to establish links between events, characters, and scenarios in diverse stories in order to construct a

event-kick-series-exploring-immigration-through-art



Figure 3: Rafael Pérez y Pérez sharing readings from “MEXICA: 20 years, 20 stories” at the Guadalajara’s International Book Fair, México.

coherent unity, but also to develop mechanisms that allow for the development of unified aesthetic intention, e.g. minimalism, through the book.

Discussion

We argue that CC systems and their products will benefit from being analyzed and evaluated by people outside of our community. We refer to this task as *computational creativity field work*. Although some colleagues have already moved in that direction (Twitter Bots (Veale 2015), DARCI (Norton, Heath, and Ventura 2015), The Painting Fool (Colton 2012), Improvisor (Keller and Morrison 2007), etc.) we have found it hard to come across papers that reflect on and share insights about such experiences; as a result, our community is missing relevant knowledge.

We have used our experiences with Alysia and MEXICA to illustrate how this field work might operate and the kind of knowledge that we can gain from it. In the following we point out some of the relevant aspects that surfaced from such practices:

- As we have showed in the previous sections, sometimes researchers see their projects only from one specific point of view; we refer to this as “researcher fixation.” Perspectives from experts and artists in other fields about our systems and products might help to prevent such fixation and trace future research paths.

Making a system or its artifacts widely accessible naturally gives rise to expert feedback. For instance, we were surprised that some colleagues in social sciences saw a research potential of our systems in their field. This may well lead to exploring entirely novel applications for our models.

- A large number of users using a co-creative system can lead to effective means of evaluating the process, through both user feedback (solicited and unsolicited) and the analysis of user data. We notice that unsolicited feedback was a particularly useful source of information.
- Creative agents capable of representing social concerns might contribute to the study of creativity and to the study of such social phenomena. Furthermore, we strongly believe that, as scientists, we have a responsibility to at least reflect on the social implications of our projects. Similarly, we need to think how our systems might contribute to society. For instance, the reader might imagine how Alysia might help to preserve and spread (part of) the musical tradition of the original inhabitants of the south of México.
- We need to provide broader audiences with artefacts that they feel comfortable with. This is a vital point that needs to be taken seriously. We cannot expect broader audiences to interact with systems that lack accessible user interfaces, or to engage with information or artifacts published through traditional academic channels. The channel should ideally represent the manner in which broad audiences are used to receiving artifacts in the domain, such as books for stories and narrative, gallery showing for art, and concerts for music. Feedback on the process of our co-creative systems is best enabled by making them available through easily-accessible channels (website, App store, etc.) paired with a simple user interface.
- Our systems received different kinds of reactions. We found that Alysia's feedback was more specific, targeting precise features of the agent. Users wanted the system to provide further assistance and to be better tailored to their needs. The co-creative aims for Alysia make this type of feedback most beneficial to further development of the system. By contrast, MEXICA's feedback was more general. Expert feedback helped to frame MEXICA in a social context as well provide valuable insight for directions for future work.

In order to make the most of practices such as those described here, our community needs to develop methodology for its field work. It is out of the scope of this paper to define such methodology; however, we would like to contribute with four initial ideas.

1. Classifications. The methodology should classify the human actors participating in field work. We distinguish the following categorizations:
 - The interdisciplinary team working on a specific project.

- Experts from other disciplines that are unfamiliar with the goals and methods of CC.
- Lay audiences outside of academia.

The last two categories conform what we refer to as "broad audiences." We expect to obtain different information from the last two types of audiences.

In the same way, we suggest to classify CC artefacts into at least three categories:

- Co-creative systems and their products
- Independent systems and their products
- Products of creative systems

This classification can be used in conjunction with the Computational Creativity Continuum (Pérez y Pérez 2018) in order to organize the information obtained from broad audiences. For instance, we can compare the feedback that mathematical/engineering oriented co-creative systems receive against the comments that cognitive-oriented co-creative systems receive. Of course, there are other possible classifications.

2. Collection of data. There are several ways to collect data. In this text, log files and unsolicited feedback stand out. Implicit feedback is particularly useful due to its utility in identifying potential pitfalls in the co-creative process, discover which components are most used (and as such potentially most successful) and identify whether the system achieves its objective of improving human creativity in an engaging manner through retention metrics.

Unsolicited feedback proved to be a valuable source of information. A subject that takes the initiative of providing comments about her experience with CC artefacts clearly is a motivated person that has engaged with such an artifact. Unsolicited comments might come, for instance, from direct messages from a user (email, social messaging, etc). But they also can take the form of interviews, essays, reviews, and so on, pondering a system and/or its products. In this case, besides having an inspired subject, we also need a person capable of making sense, from their own perspective, of the creative agent and its outputs. A good methodology should consider these and other possibilities in order to gain as much insight as possible.

3. How to present a system and/or its products to a broader audience requires a detailed study. We need to create situations where people feel comfortable; encourage the development of products designed to work as bridges with other disciplines; analyze which outputs are better suited to represent particular outputs. The main drawback is the resource-consuming nature of making systems and artifacts broadly accessible. How do we make CC systems and/or their artifacts broadly accessible in a repeatable, effective manner? Is there a minimum accessibility metric that is sufficient to gain broad audience input? Is it

possible to create a joint mechanism for making CC systems (perhaps across a similar domain) broadly accessible without the associated overhead for each individual researcher?

4. Evaluation. As part of the methodology, we suggest to break down evaluation techniques into internal and external. Internal evaluation criteria would include surveys we design and run on our students or CC researchers, and other evaluation methods that occur within the community. External criteria could include the opinion of domain experts out of CC (literature expert, painter, musician, etc.), or the utility of a co-creative system, or artifact created by a CC system, for the general population.

Internal evaluation has the indisputable advantage of scientific rigor. We have established criteria for what constitutes a legitimate survey, how questions should be phrased, and mechanisms to avoid biasing the subjects. By contrast, external evaluation can be unpredictable and difficult to control. This can help push the researcher outside of the “Hans Horse” phenomenon⁹, where we unintentionally overestimate the capabilities of systems we create.

We hope this paper will encourage the CC community to participate in the development of a methodology and engagement in field work. Ultimately, research is about venturing out into the unknown and discovering that which has never been previously found through whatever means necessary. This means different things for different research projects. There are many different dimensions and methods for exploration, where taking CC outside of CC is just one of many possibilities.

References

Ackerman, M., and Loker, D. 2017. Algorithmic song-writing with Alysia. In *International Conference on Evolutionary and Biologically Inspired Music and Art*, 1–16. Springer.

Colton, S., and Ventura, D. 2014. You can’t know my mind: A festival of computational creativity. In *ICCC*, 351–354.

Colton, S.; Llano, T.; Hepworth, R.; Charnley, J.; Gale, C.; Baron, A.; Pachat, F.; Roy, P.; Gervás, P.; Collins, N.; et al. 2016. The beyond the fence musical and computer says show documentary.

Colton, S. 2012. The painting fool: Stories from building an automated painter. In *Computers and creativity*. Springer. 3–38.

Cope, D., and Mayer, M. J. 1996. *Experiments in musical intelligence*, volume 12. AR editions Madison.

Fitch, A. 2018. The computational and the cognitive-social: Talking to Rafael Pérez y Pérez. Blog Los Angeles Review of Books.

⁹The parable of a man who fooled himself into believing that his horse could do arithmetic, when in fact it was merely attuned to its owner’s facial cues.

Irani, L.; Vertesi, J.; Dourish, P.; Philip, K.; and Grinter, R. E. 2010. Postcolonial computing: a lens on design and development. In *Proceedings of the SIGCHI conference on human factors in computing systems*, 1311–1320. ACM.

Jordanous, A. 2012. A standardised procedure for evaluating creative systems: Computational creativity evaluation based on what it is to be creative. *Cognitive Computation* 4(3):246–279.

Keller, R. M., and Morrison, D. R. 2007. A grammatical approach to automatic improvisation. In *Proceedings, Fourth Sound and Music Conference, Lefkada, Greece, July*. “Most of the soloists at Birdland had to wait for Parker’s next record in order to find out what to play next. What will they do now.

Norton, D.; Heath, D.; and Ventura, D. 2015. Accounting for bias in the evaluation of creative computational systems: An assessment of darci. In *ICCC*, 31–38.

Pérez y Pérez, R., and Sharples, M. 2001. Mexica: A computer model of a cognitive account of creative writing. *Journal of Experimental & Theoretical Artificial Intelligence* 13(2):119–139.

Pérez y Pérez, R., and Sharples, M. 2004. Three computer-based models of storytelling: Brutus, minstrel and mexica. *Knowledge-based systems* 17(1):15–29.

Pérez y Pérez, R. 2014. The three layers evaluation model for computer-generated plots. In *ICCC*, 220–229.

Pérez y Pérez, R. 2015a. A computer-based model for collaborative narrative generation. *Cognitive Systems Research* 36:30–48.

Pérez y Pérez, R. 2015b. From mexica to mexica-impro: The evolution of a computer model for plot generation. In *Computational Creativity Research: Towards Creative Machines*. Springer. 267–284.

Pérez y Pérez, R. 2015c. Mexica-impro: Generación automática de narrativas colectivas. In y Pérez, R. P., ed., *Creatividad Computacional*. México City: Grupo Editorial Patria.

Pérez y Pérez, R. 2017. *Mexica: 20 Years-20 Stories*. Counterpath.

Pérez y Pérez, R. 2018. The computational creativity continuum. In *Proceedings of the Ninth International Conference on Computational Creativity ICC*, 177–184.

Veale, T. 2015. Hallando la creatividad en conceptos en conflicto. In y Pérez, R. P., ed., *Creatividad Computacional*. México City: Grupo Editorial Patria.

The Draw-A-Computational-Creativity-Researcher Test (DACCRT): Exploring Stereotypic Images and Descriptions of Computational Creativity

Sarah Harmon and Katie McDonough

Computer Science Department

Bowdoin College

Brunswick, ME 04011 USA

{sharmon, kamcdono}@bowdoin.edu

Abstract

Prior work investigating student perceptions of scientists has revealed commonly-held beliefs, stereotypes, and even connections to career choices. We adapt the “Draw-A-Scientist” instrument to examine how undergraduates depict computational creativity researchers and the field of computational creativity as a whole. Our results indicate that there are significant differences when students are asked to draw or describe a computer scientist versus a computational creativity researcher. Whether the student is an upper-level or introductory computer science student appears to also influence responses.

Introduction

Computational creativity is a blossoming field that may help students - perhaps even traditionally underrepresented groups of students - view computer science through an interdisciplinary lens. Recently, scholars have sought to organize available resources and fundamental concepts toward the development of modern, standardized pedagogical approaches for computational creativity education (Ackerman et al. 2017). As we continue to perfect how we teach computational creativity, it is important to study and seek to understand how students perceive the field and its researchers, and how they convey these ideas to others. If misconceptions or stereotypes exist, we must first identify their presence and root causes if we wish to address them in the classroom or within research contexts.

The contribution of this work is to begin the first step toward that process. We adapt a popular survey instrument (the “Draw-A-Scientist” Test) to assess undergraduate perceptions of computational creativity research.

Related Work

In 1957, a nationwide survey was presented to high school students across the United States. Survey administrators at over 120 schools asked students to write an essay about what they thought about science and scientists. As a result, a stereotypic image of a scientist was revealed, with positive and negative aspects. For example:

The scientist is a man who wears a white coat and works in a laboratory. He is elderly or middle aged



Figure 1: A typical response to the DACST drawing prompt by an undergraduate student in an introductory computer science course.

and wears glasses. He may be bald. He may wear a beard, may be unshaven and unkempt...a very intelligent man...he works for long hours in the laboratory, sometimes day and night, going without food and sleep (Mead and Metraux 1957).

This study, along with consistently-stereotypical presentations of scientists in the media, spurred the development of the “Draw-A-Scientist” Test (DAST): an instrument used to analyze how individuals perceive science and scientists (Chambers 1983). Often, DAST investigations have generally sought to identify positive and negative indicators on similar drawing or descriptive tasks. For example, images are often annotated as positive if the depicted individual is smiling (Nuno 1998).

The DAST instrument has been adapted in several instances to better understand stereotypic images of engineering and computer science in particular. For example, the “Draw-An-Engineer” Test (DAET) has been proposed to help assess students’ ideas about engineering (Knight and Cunningham 2004; Ganesh et al. 2009; Dyehouse et al. 2011). In 2004, Martin conducted a like-minded study toward analyzing how college students perceive computer sci-

entists and the nature of their work. First-years in an introductory computer science course were asked to answer the question “What is computer science?” and to draw a picture of a computer scientist. Martin found that “all of the drawings depict[ed] white males in various degrees of geekiness” regardless of participant gender, and concluded that “CS has a fundamental image problem” (Martin 2004). More recently, Hansen et al. presented the “Draw-A-Computer-Scientist” Test (DACST) as a means to better understand elementary (fourth through sixth grade) student perceptions of computer scientists and the field of computer science. Similar findings arose with respect to gender: “71% of students drew a male computer scientist, while only 27% drew a female computer scientist”. Furthermore, 90% of computer scientists were depicted as working alone, and 82% of students included a computer as part of the drawing. When asked to describe their drawings, the most frequent words students employed were “working (23%), coding (18%), making (16%), typing (9%), doing (7%), looking (7%), fixing (7%), and testing (6%)”. The computer scientists themselves were described as working on vague tasks, and essentially were represented as “scientists who use computers” (Hansen et al. 2017).

Based on these findings, we believe that there may be similar, or at least related, stereotypic images and descriptions regarding what computational creativity research is and what a computational creativity researcher looks like. A fundamental understanding of these ideas, as well as whether computational creativity also has an “image problem”, is important to explore if we wish to enhance the perception of computational creativity as an inclusive field appropriate for all.

The “Draw-A-Computational-Creativity-Researcher” Test (DACCRT)

To investigate perceptions of computational creativity as a field, we adapted the DACST by replacing instances of “computer scientist” with “computational creativity researcher”. We also added several questions asking students to define the field and provide their thoughts about its usage in society.

Our adapted instrument has seven items, which are presented as follows:

1. Close your eyes and imagine a computational creativity researcher at work. Then, open your eyes. In the box provided below, draw what you imagined.
2. Describe what the computational creativity researcher is doing in your picture. Write at least two sentences.
3. List at least three words or phrases that come to mind when you think of this researcher.
4. What kinds of things do you think this researcher does on a typical day? List at least three things.
5. In your own words, define *computational creativity*.
6. Do you think computational creativity is generally beneficial for society? Why or why not?

7. Do you know someone who works in computational creativity research (Yes/No)? If yes, then who are they?

We will hereafter refer to this adapted instrument as the DACCRT (“Draw-A-Computational-Creativity-Researcher” Test). The following sections will describe our preliminary results of administering this instrument (and an adapted form of DACST serving as its parallel for comparison purposes) to undergraduate students.

Method

Overall, 96 undergraduate students consented to participating in the study. Of these, 31 introductory-level CS students (12F, 18M, 1 declined to state gender identity) completed the DACST. 65 students completed the DACCRT, including 29 (15F, 14M) introductory-level and 36 (12F, 24M) upper-level computer science students. In all cases, it is important to note that the survey instruments were administered by a female professor who is both a computer scientist and computational creativity researcher. This professor teaches traditional computer science courses and the only course on computational creativity in addition to supervising upper-level (undergraduate) projects in computational creativity at the institution where the survey was administered. None of the students surveyed in this report had completed the computational creativity course at this institution or participated in any computational creativity work with this professor prior to completing the questionnaire, but they may have associated the professor with computer science, computational creativity, or both.

Participants were informed of the study procedures, but there were no formal discussions of research, computer science, or computational creativity in any condition directly prior to completing the questionnaire. It was explained that the purpose of the research study was to gather information from students regarding their perceptions about computer science research, and that this study may help us gain a better understanding of the ways in which students think about computer science research as part of their studies or as a career path. Participants were also informed their responses were anonymous and voluntary, and that they were welcome to stop or skip questions at any time without consequence.

Based on past work (Chambers 1983; Hansen et al. 2017), we chose the following as major drawing indicators of the standard image of a scientist:

- Lab coat, closed toe shoes, eyeglasses, goggles, gloves (*stereotypical wearables*)
- Scientific instruments, lab equipment including beakers, whiteboard/blackboard, equations, mathematical symbols (*symbols of research*)
- Books, reports or other papers, filing cabinets (*symbols of knowledge*)

Similarly, the following were considered indicators of an artist (Kelly 1999; Kindler, Darras, and Kuo 2003):

- Radio, music notes, musical instruments, singing, music software (*symbols of music*)

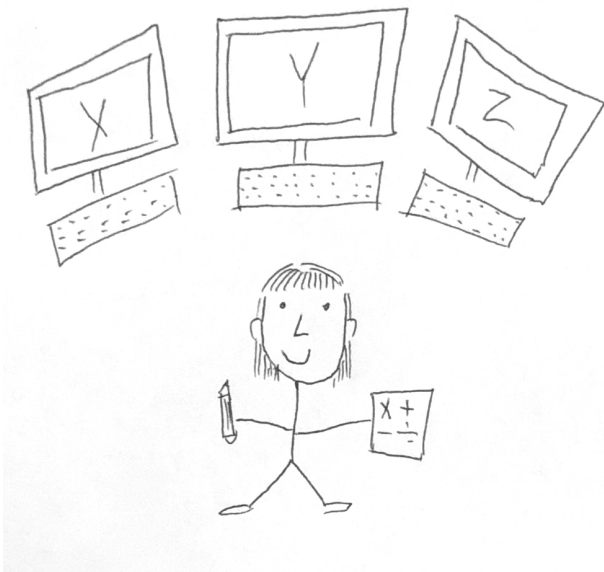


Figure 2: An introductory student drew this smiling computational creativity researcher with scientist indicators (mathematical symbols) and multiple computers.

- Beret, smock, palette, easel, paintbrush, visual artwork, dark clothes, cigarette, digital art software, sculpture, photography (*symbols of visual arts*)
- Poetry, literature, quill, ink (*symbols of literary arts*)
- Dance, stage, video camera, clapperboard, microphone (*symbols of performing arts*)

We hypothesized that students who were asked to draw a computational creativity researcher would include both scientist and artist indicators, while those who were asked to draw a computer scientist would only include scientist indicators, if at all. We expected mostly positive responses overall due to the potential effects of social desirability bias, although we hypothesized that there may be a small amount of responses that contain markers of poor health or work/life balance (drinking coffee, being tired, messiness, etc.) as described by Martin (Martin 2004). We also expected the majority of the depictions to be of an individual working alone as opposed to collaborating with others as suggested by prior work. Additionally, if gendered, the individual was hypothesized to be more frequently gendered male in all cases (Hansen et al. 2017).

Results

Computers and Collaboration

90.3% of the DACST group drew a computer in their picture, while only 6.5% indicated the person was collaborating. Among the DACCRT group who was new to computer science, 89.7% drew a computer in their picture, and 17.2% were collaborating with others. Finally, among the upper-level computer science students who completed the DACCRT, 55.5% included a computer in their drawing and 13.8%

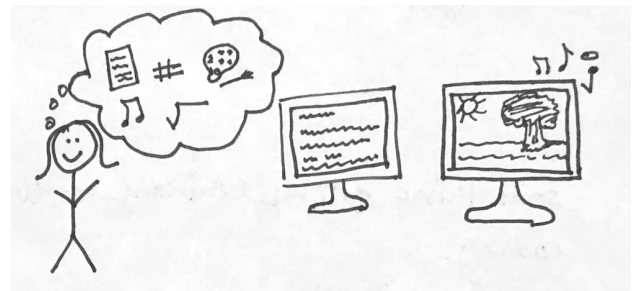


Figure 3: An upper-level student drew this smiling computational creativity researcher with artist and scientist indicators (symbols of visual art, music, and research) alongside two computers.

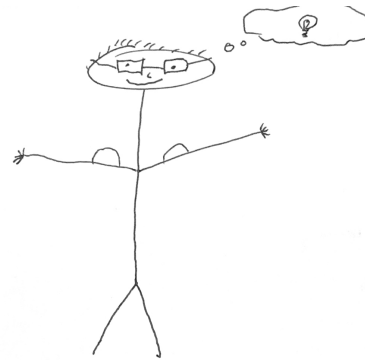


Figure 4: A positive depiction of a computational creativity researcher as a scientist who is simply thinking of an idea, with well-toned arms stretched wide.

drew a researcher that was collaborating with others. To easily compare these results across conditions, refer to Table 1. No statistical significance was found between levels of collaboration displayed. However, a two-tailed Fisher's exact test revealed a high statistical significance between the DACST and the upper-level DACCRT ($p = 0.0024$) as well as the introductory versus the upper-level DACCRT conditions ($p = 0.0029$) in terms of whether a computer was present.

One individual was depicted as using multiple computers 6.5%, 20.7%, and 5.6% of the time across the DACST, introductory-level DACCRT, and upper-level DACCRT conditions. No statistical significance was observed across conditions.

Pronoun Usage

We examined the pronoun usage in the drawing descriptions to determine if there were noticeable differences (Table 2). The difference in he/him/his and they/them/theirs pronoun usage was found to be statistically significant using a two-tailed Fisher's exact test between the two DACCRT groups ($p = 0.0187$; $p = 0.0204$) as well as between the upper-level DACCRT and the DACST group ($p = 0.0359$; $p = 0.0493$),



Figure 5: A drawing that shows a computational creativity researcher coding at night “because CS people have bad sleep schedules”, with code producing music, visual art, and literature.

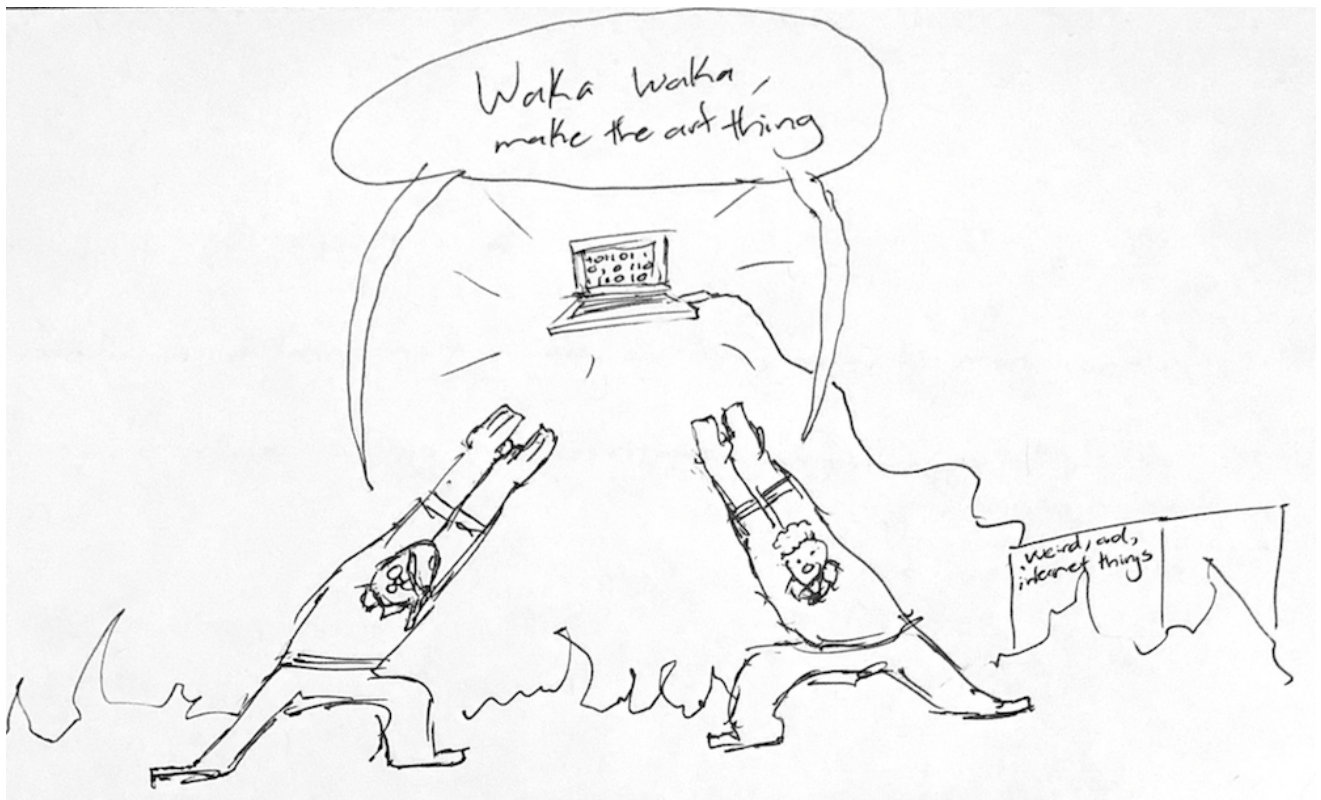


Figure 6: An example depiction of computational creativity researchers collaborating, as drawn by a computer science major.

	Computer	Alone	Smiling
DACST (introductory)	90.3%	93.5%	12.9%
DACCRT (introductory)	89.7%	82.8%	20.7%
DACCRT (upper-level)	55.5%	86.2%	36.1%

Table 1: Presence of a computer, a lone, non-collaborating individual, and a smiling individual across the three drawing response experimental conditions. The most prominent finding is highlighted in boldface: the fact that participants in the upper-level DACCRT conditions appeared to be less likely to include a computer in their drawing.

respectively. No statistical significance was observed between any other conditions.

Smiling as a Positive Indicator

12.9% drew smiling computer scientists in the DACST group, while 20.7% and 36.1% of the depicted researchers were smiling in the introductory and upper-level DACCRT groups, respectively (Table 1). No statistical significance was found between the two introductory-level groups or the two DACCRT groups. The difference between the DACST and the DACCRT upper-level group was found to be statistically significant using a two-tailed Fisher's exact test ($p = 0.0475$). No statistical significance was found between smiling and pronoun usage.

Artist and Scientist Indicators

The presence of scientist and artist indicators are shown in Table 3. In the DACST group, none of the drawings displayed artist indicators, but 12.9% of the drawings displayed scientist indicators.

In the introductory-level DACCRT condition, 27.6% used scientist indicators, while 24.1% used artist indicators. Interestingly, scientist and artist indicators were almost never combined in a drawing. More often, a computer was simply drawn along with either a scientist indicator or an artist indicator. Only 3.4% of the introductory DACCRT participants included both types of indicators at once.

Among the upper-level computer science students who completed the DACCRT, 47.2% included artist indicators, and the same percentage included scientist indicators. 19.4% included both.

No statistical significance was found between the introductory or the DACCRT groups in terms of scientist indicators. In contrast, high statistical significance was observed using a two-tailed Fisher's exact test between the DACST and upper-level DACCRT conditions ($p = 0.0034$).

There was no statistically significant difference found between the DACCRT groups in terms of artist indicators. However, high statistical significance was found using a two-tailed Fisher's exact test between the DACST and introductory-level DACCRT conditions, and between the DACST and upper-level DACCRT conditions based on the presence of artist indicators ($p = 0.0040$ and $p < 0.0001$, respectively).

In terms of displaying both scientist and artist indicators, statistical significance was observed using a two-tailed Fisher's exact test between the DACST and upper-level DACCRT conditions ($p = 0.0126$). There was no statistical significance found between the two introductory or the two DACCRT conditions.

Word Associations

In the DACST group, the most frequent word used was "smart", occurring in 25.8% of responses. Among the introductory and upper-level DACCRT conditions, the most common word used was "creative" (20.7% and 27.8% of responses, respectively). Most word associations were positive or neutral, with only a few indicating negative connotations (e.g., "tired", "frustration", "overworks", "wired/tense", "unorganized"). The most frequent task that a computational creativity researcher does on a given day was "coding", appearing in 62.1% and 36.1% responses for the introductory and upper-level DACCRT conditions, respectively.

Purpose, Meaning, and Impacts of Computational Creativity

Among the introductory-level DACCRT participants, 86.2% answered that computational creativity was beneficial for society. 10.3% indicated computational creativity was probably beneficial, but they were not confident about their own definition. Finally, 3.4% stated that there were advantages and drawbacks. Similar statistics were seen for the upper-level DACCRT participants. Participant responses were annotated as *Yes* 75% of the time, *Probably, just not sure what it is* 16.7% of the time, and *Yes and No* 8.3% of the time.

Stated disadvantages of pursuing computational creativity included taking away jobs from creative individuals or integrating too much technology into daily life so that humans would become too dependent on it. Generally, however, computational creativity was often described as beneficial because it allows us to find innovative solutions (*innovation*) or find solutions quickly (*efficiency*) to complex problems. Other responses included the idea that it helps us broaden what we can accomplish with technology or offer new perspectives (*enlightenment*), helps individuals who do not work with technology to understand the technology in their lives (*inclusion*), enables people to express themselves (*self-expression*), or could influence others in society (*impact*).

Health and Life/Work Balance

In each condition, a small percentage of responses drew or described the researcher as pursuing unhealthy work/life habits (3.3%, 6.9%, and 8.3% of the DACST group, the introductory DACCRT, and the upper-level DACCRT conditions, respectively). This was often depicted or described in terms of drinking coffee, dark circles under eyes, and not getting enough sleep (as in Figure 5). No statistical significance was found between any of the conditions.

	No Pronouns Specified	They/Them/Theirs	S/he or He/She	She/Her/Hers	He/Him/His
DACST (introductory)	35.5%	25.8%	12.9%	0.0%	25.8%
DACCRT (introductory)	34.5%	20.7%	6.9%	10.3%	27.6%
DACCRT (upper-level)	22.2%	50.0%	8.3%	13.9%	5.6%

Table 2: Pronoun usage by experimental condition. The most prominent findings are highlighted in boldface: the upper-level DACCRT students appeared to be more likely to use they/them/theirs and less likely to use he/him/his.

	Scientist	Artist	Both
DACST (introductory)	12.9%	0.0%	0.0%
DACCRT (introductory)	27.6%	24.1%	3.4%
DACCRT (upper-level)	47.2%	47.2%	19.4%

Table 3: Presence of scientist and artist drawing indicators across each of the experimental conditions. The most prominent finding is highlighted in boldface: the fact that participants in the DACST condition appeared to be less likely to include one or more artist indicators in their drawing.



Figure 7: Responses to the DACCRT by upper-level students sometimes resulted in “creative” depictions of a computational creativity researcher such as this one.

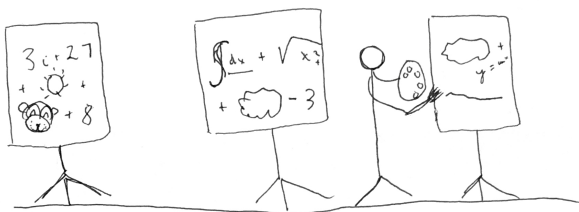


Figure 8: A depiction of a computational creativity researcher painting mathematical and artistic symbols onto multiple easels. This was not the only example in which the researcher was painting with these kinds of symbols. For instance, another drawing from an introductory-level computer science student featured a researcher painting with a palette of colors labeled as *life experience*, *research*, *science*, *art*, and *math*.

Discussion

This work examined the depictions of computer scientists and computational creativity researchers by undergraduate students in computer science courses. As such, it serves as a first step toward understanding how computational creativity is portrayed as a field of study and potential career path.

Due to space constraints, we leave a more detailed discussion of identity (e.g. gender identity) as it relates to this topic and participant understanding of computational creativity for future work. Additionally, although it was explained to participants that the purpose of the study was to better understand perceptions of computer science *research*, the differences between being prompted to draw a *scientist* and a *researcher* should be more thoroughly explored. It would be interesting to inquire, for instance, what a *computational creativity scientist* versus a *computer researcher* (or a *computer science researcher*) looks like. Surveying individuals who are not just computer science undergraduate students, as well as to survey a larger population in general (so to more precisely examine relationships between the factors discussed in this work) would likewise be valuable.

Although the majority of computational creativity researchers were depicted as alone, some descriptions indicated students perceived these individuals as highly collaborative. For example, one computer science major remarked “My researcher is just waving hello at their coworkers as CC seems like a very collaborative field”, while an introductory student explained “I added another person because I imagine this to be collaborative because of the word creativity”. Even if another person was not depicted in the drawing, the quality of being friendly and open may have been depicted through other factors, such as body language. For example, one upper-level student described their researcher as “They’re smiling and thinking. Mostly just happily standing, arms stretched wide” (Figure 4). Due to these findings, it may be worth investigating the perceived relationship between computational creativity and collaboration and openness toward others, even though we did not observe any statistically-significant associations in this work.

One striking result is that upper-level students were significantly less inclined to include a computer as part of their depiction of a computational creativity researcher. Participants in general presented a greater variety of depictions of what CC researchers might be doing. This result might be attributed to the fact that the participants were primed with the word “creativity” and thus sought to draw a more inventive picture. However, the introductory-level DACCRT group included a computer as part of their drawing almost as much as the DACST group. Instead, then, it is possible that

as students gain experience in computer science, they begin to recognize that performing research in a computer science field does not always involve a computer. As an alternative explanation, upper-level students might also be more likely to use devices such as metaphorical symbols as part of their drawing. To illustrate, several examples are shown in Figures 6 and 7.

Interestingly, even upper-level computer science students sometimes drew a computational creativity researcher as a scientist who uses computers, similar to Hansen et al.'s work with elementary school students (Hansen et al. 2017). Some depictions, for instance, included individuals with explicitly-labeled lab coats and closed-toed shoes. We did not expect this kind of depiction as strongly from college students who had already taken several computer science courses. This result may be because they are not necessarily drawing their own perceptions of computer scientists and computational creativity researchers, but their perceptions of the extreme end of stereotypes known to society. They may value developing a recognizable depiction for others as opposed to producing a more realistic image that someone else might not connect with.

Our findings further suggest that being asked to draw a computational creativity researcher as opposed to a computer scientist potentially leads to the inclusion of artist indicators. In one depiction by an upper-level student, a researcher is holding a palette and painting mathematical symbols alongside visual art on three easels, with not a computer screen in sight (Figure 8). Whereas a computer scientist might be depicted with multiple computers, then, sometimes a computational creativity researcher is depicted with multiple easels! Perhaps this multiplicity is sometimes meant to evoke a sense of complexity or quantity of work produced, or possibly a true dedication to one's work, similar to that revealed in the original stereotypical image for a scientist (Mead and Metraux 1957).

The inclusion of artist indicators in general is possibly due to the connotations of the term "creativity", but this idea should be verified in future work. Perhaps even more intriguing is the fact that students appeared to classify a computational creativity researcher as being either strongly a scientist or an artist, but not both (as frequently). This may point to an underlying belief in society that science and art cannot be combined. Computer scientists, in contrast, are commonly described using scientist indicators or simply as being next to a computer. The stereotypic image of a computer scientist appears to be one who does not engage in artistic endeavors, but simply one who remains with "eyes glued to a computer monitor" (Martin 2004).

Finally, as few but measurable indications of poor work/life balance were observed across all three conditions, we encourage our peers to work toward healthy lifestyles, and to help their students and lab associates do the same. Let's keep smiling, with our arms stretched wide.

References

Ackerman, M.; Goel, A.; Johnson, C. G.; Jordanous, A.; León, C.; y Pérez, R. P.; Toivonen, H.; and Ventura, D. 2017. Teaching computational creativity. In *Proceedings of*

the Eighth International Conference on Computational Creativity, 9–16.

Chambers, D. W. 1983. Stereotypic images of the scientist: The draw-a-scientist test. *Science education* 67(2):255–265.

Dyehouse, M.; Weber, N.; Kharchenko, O.; Duncan, D.; Strobel, J.; and Diefes-Dux, H. 2011. Measuring pupils perceptions of engineers: Validation of the draw-an-engineer (daet) coding system with interview triangulation. *Research in Engineering Education Symposium*.

Ganesh, T.; Thieken, J.; Elser, M.; Baker, D.; Krause, S.; Roberts, C.; Kurpius-Robinson, S.; Middleton, J.; and Golden, J. 2009. Eliciting underserved middle-school youths' notions of engineers: Draw an engineer. In *ASEE Annual Conference and Exposition, Conference Proceedings*.

Hansen, A. K.; Dwyer, H. A.; Iveland, A.; Talesfore, M.; Wright, L.; Harlow, D. B.; and Franklin, D. 2017. Assessing children's understanding of the work of computer scientists: The draw-a-computer-scientist test. In *Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education*, 279–284. ACM.

Kelly, A. 1999. Art-science connections: An investigation of creative innovation.

Kindler, A. M.; Darras, B.; and Kuo, A. C. S. 2003. An artist: A cross-cultural or a culture-specific category? *The International Journal of Arts Education* 1(1):94–117.

Knight, M., and Cunningham, C. 2004. Draw an engineer test (daet): Development of a tool to investigate students ideas about engineers and engineering. In *ASEE Annual Conference and Exposition*, volume 2004.

Martin, C. D. 2004. Draw a computer scientist. *ACM SIGCSE Bulletin* 36(4):11–12.

Mead, M., and Metraux, R. 1957. Image of the scientist among high-school students. *Science* 126(3270):384–390.

Nuno, J. 1998. Draw a scientist: Middle school and high school students conceptions about scientists. *Retrieved on June 11:2008*.

The Importance of Applying Computational Creativity to Scientific and Mathematical Domains

Alison Pease¹, Simon Colton², Chris Warburton¹, Athanasios Nathanail³,
Irina Preda¹, Daniel Arnold³, Daniel Winterstein¹ and Mike Cook²

¹University of Dundee, UK

²Queen Mary University of London, UK

³Heriot-Watt, University of Dundee, UK

Abstract

Science and mathematics are currently under-represented in the computational creativity (CC) community. We discuss why the CC community should apply their work to mathematical and scientific domains, and argue that this would be mutually beneficial for the domains in question. We identify a key challenge in Automated Reasoning – that it has not achieved widespread adoption by mathematicians; and one in Automated Scientific Discovery – the need for communicability of automatically generated scientific knowledge. We recommend that CC researchers help to address these two challenges by: (i) applying systems based on cognitive mechanisms to scientific and mathematical domains; (ii) employing experience in building and evaluating interactive systems to this context; and (iii) using expertise in automatically producing framing functionality to enhance the communicability of automatically generated scientific knowledge.

Introduction

Despite the best efforts of successive ICCO organising committees and the computational creativity (CC) community, CC has always attracted substantially more interest from researchers in artistic than scientific and mathematical domains. In their 2017 study of application domains in CC, (Loughran and O’Neill 2017) found that, of 16 categories, papers on Maths, Science and Logic accounted for only 3% of the 353 papers on CC across 12 years. Of course, some work is domain independent, or at least not easily assigned to an academic discipline, such as the body of work on CC and curiosity (for instance (Grace et al. 2017b)). Even taking this into account, it is clear that science and mathematics are vastly under-represented in our community.

There are many reasons why this may be the case. Firstly, AI researchers in scientific domains may well be doing creativity-related work in other contexts but not engaging with the CC community. Automated reasoning (usually deductive reasoning in mathematics) and automated scientific discovery (usually inductive reasoning in a scientific domain) are both thriving subfields of AI, with internationally recognised journals as outlets for publication and engagement; certainly these will contain work relevant to our field but couched in different terminology with different methodologies. Secondly, other priorities in scientific domains may

have led to a focus on techniques such as search, data-mining and automated deduction. Since these generate results of interest to domain experts, the more difficult, fluid and tenuous concept of creativity may be seen as unnecessary, risky or simply not a priority. This may particularly be the case given the various “AI winters” in the twentieth century (the second of which ended in 1993, just six years before the first workshop on CC), and the need for AI to “prove itself” (Crevier 1993). Thirdly, it may be easier to be a hobbyist game designer or artist or composer (many CC researchers are deeply involved in the domains in which they work), than an AI researcher and also an occasional physicist. Fourthly, CC researchers may consider that even if generation is possible within scientific domains, evaluation is too difficult. How we should evaluate our work and our systems has always been a contentious, albeit important, issue in CC, with few proposed evaluation metrics and the majority of researchers still arguing for value along the lines of “we/people liked the system’s output” or “we/people couldn’t distinguish the system’s output from human produced work” (Jordanous 2014). It might be the case that in science, the main evaluation metric – “is it true?”, or “does it work?” – is considered simply too expensive or difficult to demonstrate. Even if evaluation is possible, we may be more prone to dismissing initial results as uninteresting in science than in artistic domains. For instance, we may get a greater sense of progress from a system working in game design which outputs a new (rather basic) game, than one working in geology which outputs a new (rather basic) result.

In this position paper we argue that neglecting scientific and mathematics domains in CC is at best a wasted opportunity, and at worst a significant problem for the field. Deep learning and ML are making inroads everywhere: generative arts, processors, Go, machine vision, and so on, and we need to consider as a community where this leaves us. We believe that it is essential to the health of our field that we reach out as a community at this stage, both to domain experts in science and mathematics and to those in related AI areas. The benefits of doing so will go both ways: we argue that research in CC can help to address key challenges in both Automated Reasoning and Automated Scientific Discovery. As AI is used more and more in science, there is greater dependence on blackbox machine learning systems. While providing greater predictive power, this often comes

at the cost of understanding. We call this the *Understandability Problem* and argue that it will become a big issue in science, which we will have to address. Twenty years of thinking about computational creativity has provided us with valuable tools for addressing these problems. This paper is a call to arms to CC researchers to apply their work to science and mathematics.

A deeper look at the 3% of papers in Maths, Science and Logic (including, for instance, (Juršič et al. 2012; O’Donoghue et al. 2015)) is outwith the scope of this paper. Similarly, we leave aside the question of what creativity or non-creativity looks like in the arts or sciences; for now we simply assume that creative endeavours take place in both domains.

Loughran and O’Neill argue that “tackling scientific, logical or realistic issues could help bring the reputation of CC away from a purely aesthetic domain towards developing solutions for real world problems.” (Loughran and O’Neill 2017, p. 7) and that “It is imperative that the field remains balanced as it grows and that we remember to reflect on all areas of growth.” (*Ibid.*). In this paper we support and present further arguments for this position, alongside practical recommendations for doing so.

What are the sciences and arts?

The concept of science is not a straightforward one. The division of the origins of learning and systematic production of new knowledge into disciplines as we know them tends to take into account at least some of the following: methodologies, objects of study (which can be shared with other disciplines), a body of accumulated knowledge (which is generally not shared with other disciplines), theories and concepts, terminology and an institutional manifesto (so that it can reproduce itself) (Krishnan 2009, p. 9). Sciences include *Natural sciences*, which are subdivided into *physical sciences* (chemistry, physics, astronomy), *life sciences*, or *biology* (zoology, botany, ecology, genetics) and *earth science* (geology, oceanography, meteorology, palaeontology); *Social sciences* (psychology, sociology, economics, law, political science); *Formal sciences* (mathematics, logic, theoretical computer science, statistics); and *Applied sciences*, which are subdivided into *engineering* (computer science, civil engineering, electrical engineering, mechanical engineering); *health sciences* (medicine, dentistry, pharmacy) and *agriculture*. The number and variety of sciences makes generalisations difficult, and core values vary accordingly. However, values commonly associated with the (rather unhelpfully named) “hard sciences” include repeatability, reproducibility, predictability, generality and understandability. This last value is particularly cherished: for instance, Roger G. Newton sums it up as “The primary aim of most physical scientists is to understand and explain the workings of Nature.” (Newton 2000, p. 4).

The arts are possibly even harder to define. Indeed, Gallie specifically uses “Art” as an example of an essentially contested concept. This is a concept, the definition of which is “not resolvable by argument of any kind” (Gallie 1955 1956, p. 169). Julie Van Camp, writing in the context of United

	Science	Arts
<i>Aesthetic:</i>	truth	beauty
<i>Approach:</i>	problem-driven	artefact-driven
<i>Task:</i>	analytic	generative
<i>Terminology:</i>	discover	create
<i>Status:</i>	objective	subjective
<i>Goal:</i>	knowledge	self-expression

Table 1: Possible perspectives on scientific and artistic endeavours.

States Congressional policy on arts education, provides the following extensional definition:

The term ‘the arts’ includes, but is not limited to, music (instrumental and vocal), dance, drama, folk art, creative writing, architecture and allied fields, painting, sculpture, photography, graphic and craft arts, industrial design, costume and fashion design, motion pictures, television, radio, film, video, tape and sound recording, the arts related to the presentation, performance, execution, and exhibition of such major art forms, all those traditional arts practiced by the diverse peoples of this country. (*sic*) and the study and application of the arts to the human environment.¹

As a starting point, we could suggest (generalising, controversially) some of the differences between the sciences and the arts as shown in Table 1. In particular, the terminological difference between discovering and creating may explain our field’s current focus on the arts. Of course, the real-world everyday lived experience of *doing* science or *doing* art is far more complex than Table 1 would suggest. Studies of interpretations of seismic data in geology, for instance, show the large number of different expert interpretations of the same seismic section, highlighting the subjectivity involved (Bond et al. 2007). These interpretations are used to analyse subsurface geology, and form the basis for many exploration and extraction decisions. Even in cases where interpreters report that an interpretation is relatively straightforward, there are significant differences in interpretation, leading to significantly different predictions, for instance about gross pore volume or gross rock volume (Rankey 2003). While objectivity may be the goal here, such studies suggest that this aspect of geological practice is closer to visual art interpretation than it is to some other scientific domains.

Similarly, studies of the backstage production of mathematics show that beauty is often a guiding value (Inglis and Aberdein 2015); there is a high level of disagreement amongst experts about the validity of certain proofs (Inglis et al. 2013); and proofs and theories are often considered to be socially constructed rather than discovered (Lakatos 1976). Less structured knowledge such as our ability to reason logically has been shown to be highly context dependent (for instance, participants in the Wason Selection Task were unable to solve a logical problem at an abstract level

¹http://web.csulb.edu/jvancamp/361_r8.html

but could solve it correctly when it was framed in a familiar context (Wason and Shapiro 1971)); constructing grounding metaphors to the physical world and abstract linking metaphors argued to be fundamental to our understanding and construction of mathematical knowledge (Lakoff and Núñez 2000); and even the language in which reasoning occurs affecting our preconceptions, perceptions and assumptions (Barton 2009). An analogous story could be told in the arts; for instance, in some contexts paintings are criticised for being beautiful, with the goal being truth, or knowledge (Derrida 1987).

Dibbets expresses the relationship between arts and sciences as follows:

But in the end, we all do very much of the same. All scientists, artists, composers and writers are intensively occupied imagining something that does not yet exist. They find themselves at the borders of areas where up to then hardly anyone found himself, trying to solve problems that are incomprehensible to others, trying to answer questions no one has ever asked. Here, they share a vision on things that are not yet real. (Dibbets 2002, p. 1)

Some of these interdisciplinary features are recognised in curriculum design and teaching featuring transferable skills, in which one skill may be learned within a scientific context and developed or employed in an arts context, or vice versa (see for instance (Gaff and Ratcliff 1996)). Of course, the need for so many interdisciplinary initiatives (and related concepts such as transdisciplinarity, pluridisciplinarity, and multidisciplinary) may suggest that some traditional discipline boundaries are no longer drawn in a helpful way. The evolving role and functionality of AI systems further complicates things. The focus of AI researchers, particularly in machine learning, is often on the skills they hope to simulate rather than a particular domain in which they are usually employed. This may be a more productive approach than the typical CC focus on domain over skill.

Automated Reasoning

Brief history

Automated Reasoning (AR) is a flourishing academic and industry community, with a range of publication venues, including the Journal of Automated Reasoning, the International Joint Conference on Automated Reasoning (IJCAR) and Conference on Automated Deduction (CADE). It has a relatively long history in Artificial Intelligence research: experiments were conducted as early as 1955, with Newell, Shaw and Simons Logic Theorist, which searched forward from axioms to look for proofs of results. Theorem provers HOL, NuPrl and Nqthm, and a variety of other approaches and software tools were in development in the mid-1980's for practical reasoning about programs: (Jones 2003) gives an account of the early history of AR. Notable recent successes include Tom Hales and his team's formalisation of their proof of the Kepler conjecture, using several theorem provers to confirm Hales' major 1998 paper (Hales et al. 2010); and Georges Gonthier and team's 2012 formalisation of the 255 page odd-order theorem (Gonthier 2013) (one of

the most important and longest proofs of 20th century algebra) in the Coq theorem prover.

Key Challenge

While the simulation of mathematical reasoning has been a driving force throughout the history of AI, *it has not achieved widespread adoption by mathematicians*. This is now seen as one of the key challenges in the field. The 2017 and 2019 Big Proof I and II Programmes² included under the Programme Theme description:

The programme is directed at the challenges of bringing proof technology into mainstream mathematical practice.³

and

The scale and sophistication of proof technology is approaching a point where it can effectively aid human mathematical creativity at all levels of expertise. (*Ibid.*)

We can hypothesise many reasons as to why there remains a disconnect between automated and human reasoning. There may be cultural reasons: mathematicians are typically not trained to use Automated Theorem Provers (ATPs), it is not usually part of the undergraduate course or subsequent training and practice. It may simply be the case that perhaps mathematicians that use AR become known as computer scientists (definitions of both of these professions are fluid and somewhat overlapping). Lastly – and this oft-cited reason is our focus here – it may be because current systems cannot do mathematics in the ways that humans do: machine proofs are often considered by mathematicians to be unclear, uninspiring and untrustworthy, as opposed to human proofs which can be deep, elegant and explanatory.

Opportunities for CC

Traditionally there have been two barriers to developing systems which produce “human-like” mathematics: firstly, it is difficult to know what this is; and secondly, it is difficult to automate (Bundy 2011; Gowers 2000). The growing interdisciplinary study of mathematical practice, started by Polya (Polya 1945) and Lakatos (Lakatos 1976), can shed light on the first of these problems. They were early advocates of the (as yet unarticulated) view that it is fruitful to look at what Hersh later termed “backstage mathematics” – the informal workings and conversations about “mathematics in the making” (as opposed to “frontstage mathematics” – textbook or publication-style “finished mathematics”) (Hersh 1991). This rapidly growing body of work is interdisciplinary to varying degrees, bridging mathematics, history, sociology, philosophy, education and cognitive science of mathematics.

²These were hosted at the Isaac Newton Institute (INI) for Mathematical Sciences (2017) and, as a follow on INI satellite event (2019) at the International Centre for Mathematical Sciences in Edinburgh, and organised by some of the most influential people in automated reasoning today: <https://www.newton.ac.uk/event/bpr> and <https://www.newton.ac.uk/event/bprw02>

³<https://www.newton.ac.uk/event/bpr>

Automated Reasoning is largely based on the traditional model of mathematics as a solitary, logic-based endeavour, largely comprising of constructing mathematical proofs. This contrasts with work in the study of mathematical practice, which recognises that mathematics largely takes place in a social context; that it involves “soft” aspects such as creativity, informal argument, error and analogy; and that mathematical knowledge comprises far more than mere proof, including definitions, examples, conjectures explanations, and so on.

Developments in the study of mathematical practice include work on visualisation, such as diagrammatic reasoning in mathematics; analogies, such as between mathematical theories and axiom sets; and mathematical concept development, such as ways to determine potential fruitfulness of rival definitions. Lakoff and colleagues (Lakoff and Núñez 2000) and Barton (Barton 2009) have explored the close connection between language and thought, and shown that images, metaphors and concept-blends used in ordinary language shape mathematical (and all other types of) thinking. At the heart of many of these analyses lies the question of what proof is for, and the recognition that it plays multiple roles; explaining, convincing, evaluating, aiding memory, and so on, complementing or replacing traditional notions of proof as a guarantee of truth). This in turn gives an alternative picture of machines as members of a mathematical community.

These developments present opportunities for researchers in CC which would help to address the second barrier in the “human-like” computing movement – that of difficulty in automation.

Recommendation 1: CC researchers who have developed systems based on cognitive mechanisms, such as concept-blending, analogies and metaphors (eg. (Veale 2012; Li et al. 2012; Baydin, Lopez De Mantaras, and Ontanon 2012; O’Donoghue and Keane 2012)) may consider applying these systems to mathematical domains.

Recommendation 2: CC researchers who have experience in building and evaluating interactive systems which enhance an expert user’s creativity (eg. (Bray and Bown 2016; A. et al. 2014; Karimi et al. 2018)) may consider conducting their work with expert mathematicians. This might, for instance, follow user-centred design, development and testing, and perhaps bridge work between AR and user mathematicians.

Automated Scientific Discovery

Brief history

Whereas AR traditionally has deduction at its heart, Automated Scientific Discovery (ASD) uses induction and abduction to make new taxonomies, laws, theories, models, predictions and explanations. Again, this endeavour started early in the history of AI, with Herbert Simon’s work in 1966 on scientific discovery and problem solving (Simon 1966) and DENDRAL, which used heuristic search to systematically evaluate all of the topologically distinct arrangements of a set of atoms within the rules of chemistry (Lindsay et

al. 1993). The BACON set of programmes were also early examples of ASD, which used rule-based induction to re-discover empirical rules in history of physics and chemistry (Langley 1979). The field became more active in the late 1970s and early 1980s, starting to use machine learning data, and moved from replicating historical events to discovering new ones, including results in astronomy, biology, chemistry, geology, graph theory, and metallurgy (see (Langley 2000) for a review). Today, widespread use of sophisticated machine learning techniques, alongside an explosion of data has led to discoveries such as metallic glass, cost-effective plastic solar cells, and drug discovery.

The use of computational systems to find patterns in scientific data is not without critics. For instance, Genevera Allen highlighted accuracy and reproducibility issues with scientific discoveries made by machine-learning techniques in her recent talk at the 2019 Annual Meeting of the American Association for the Advancement of Science (AAAS).⁴

Key Challenge

Pat Langley – one of the most influential figures in ASD – highlights the need for *communicability of automatically generated scientific knowledge*. In (Langley 2002) he reviews successful and unsuccessful ASD systems, concluding that scientists want interactive, mixed-initiative, rather than fully automated systems. Results must be communicated in language and notation which is familiar to the scientist for collaboration to be successful. He also argues this point in (Džeroski, Langley, and Todorovski 2007):

This emphasis on exchanging results makes it essential that scientific knowledge be *communicable*. (*Ibid.* p2).

Opportunities for CC

The importance and value of narratives that explain, contextualise, comment, and frame generated artefacts for public or expert consumption has been recognised in CC (Charnley, Pease, and Colton 2012), and in a human-only context is known to affect perception of creativity in artistic domains. The CC project to enable creative software to produce its own framing information is in its infancy, but it forms a fundamental part of one of the main evaluation frameworks, as the “F” part of the FACE model (Colton, Pease, and Charnley 2011), and initial work has emerged. This approach goes beyond the explainable AI movement (Došilović, Brčić, and Hlupić 2018), as it aims to show motivation, aesthetic judgements, and so on; telling the story behind the creation of an artefact. We foresee this being an increasingly important area of research in CC, with an increasing level of sophistication: from explanation to justification to argument and dialogue with a user about the value, method of production, motivation etc. behind output. How framing information should be developed is a research programme in its own right. Whilst much attention has been focused on making the outputs of ML systems more accurate and robust, there

⁴https://eurekaalert.org/pub_releases/2019-02/ru-cwt021119.php

is also a need for framing information which is more explanatory, more understandable to users and less prone to misinterpretation.

Recommendation 3: CC researchers who are developing systems which can automatically produce framing information (eg. (Grace et al. 2017a; Tomašič, Žnidaršič, and Papa 2014; Colton, Goodwin, and Veale 2012)), may consider applying them to ASD. This may perhaps in collaboration with researchers and existing systems in that field, with a focus on producing useful framing information in a scientific context.

The Understandability Problem in Science and Mathematics

The key challenges that we have identified in both AR and ASD both concern understandability in science and mathematics, and present two different approaches to the problem. Our distinction by domain is, to a certain extent, artificial, and our suggested approaches and recommendations could be used in either domain. In this section we discuss more generally the issue of understanding in science and mathematics, and what that might mean in an AI world.

Understanding in human science and mathematics

Roger Newton’s quote above about the primary aim of physical scientists being to understand and explain nature is uncontroversial, but difficult to unpack. Ever since the entirety of our collective scientific knowledge became too large for a single polymath to comprehend, we have had to outsource our understanding to others. The institutionalised ways in which trust of others’ understanding and progress is handled started with the early universities, and developed with the invention of the printing press, academic journals, the peer review process and so on. Knowledge and understanding is a social process, as argued by (Martin and Pease 2013), but even in the human-only case, this gets complicated. The longest proof in history, of the Classification of Finite Simple Groups (CFSG), is over 10,000 pages, spread across 500 or so journal articles, by over 100 different authors, and took 110 years to complete. What does understanding mean here? Perhaps a handful of people understand the proof in its entirety, and when they die it is not obvious that any one person will ever again understand the entire proof.

In the example of the CFSG, it is considered sufficient that someone once understood the proof. However in the ongoing case of the *abc* conjecture, this is not the case. This conjecture – proposed in 1985, on relationships between prime numbers – is considered to be one of the most important conjectures in number theory (more significant than Fermat’s Last Theorem; in fact Fermat’s Last Theorem would be a corollary of the proof). A proof would be “one of the most astounding achievements of mathematics of the twenty-first century.” (Goldfeld, in (Ball 2012)). In 2012 Shinichi Mochizuki – a mathematician with a good track record, having proved “extremely deep” theorems in the past (Conrad in (Ball 2012)) – produced a 500-page proof. The problem is that the techniques and mathematical objects which Mochizuki has developed to use in his

proof are so new and strange that it would take a reviewer or mathematical colleague most of their career to understand them, before they were able to understand and verify the proof. Despite some efforts from Mochizuki and a handful of his followers to make his work accessible, currently his proof has neither been published nor accepted by mainstream mathematicians, for the simple reason that they don’t understand it.

Crowd-sourced mathematics, in which open conjectures are solved collaboratively via online communities, has been used for around ten years now by a subset of the mathematical community as a new way of producing mathematics through collaboration and sharing (Gowers and Nielsen 2009). Nielsen argues that this has resulted in “amplifying collective intelligence” in his book *Reinventing Discovery* (Nielsen 2011). It has certainly resulted in some original and significant new proofs (for instance, the proofs of the Bounded Gaps Between Primes and the Bounded intervals with many primes, in the 2014 Polymath8 project (eg. (Polymath 2014))). Here it is perfectly possible for a person to be a co-author but not fully understand the proof in their own paper.

Understanding in mixed-initiative science and mathematics

Adding computers to the social process, to form a combination of people, computers, and mathematical archives to create and apply mathematics – a “mathematics social machine” (Martin and Pease 2013) – further complicates matters. Take *automated theorem proving*; the task of deciding whether a given formal statement follows from a given set of premises (Sutcliffe and Suttner 2001). The least informative approach would be to produce merely a “yes”, “no” or “unknown” response. Not only is this devoid of *explanation*, but it also hides the effects of any bugs; requiring the user to either trust the results, or verify the implementation.

This can be mitigated by having the system instead generate a *proof object*: a formal argument for *why* a given statement follows or does not follow. Once generated, a proof object’s validity can be checked without requiring any knowledge of how it was created, thus avoiding the need to trust or verify the (potentially complicated) search and generation procedures. Theorem provers which produce proof objects that are trivial to check by independent *proof checker* programs (which are themselves easily verified, due to their simplicity) satisfy the *de Bruijn criterion* (Barendregt and Wiedijk 2005); examples are Coq (Barras et al. 1997) and Isabelle/HOL (Nipkow, Paulson, and Wenzel 2002).

Proof objects are not a complete solution to understandability, since they can still be quite inscrutable to human users. This often depends on how closely the chosen formal system is able to encode the user’s ideas: for example, the formal proof of the Kepler Conjecture was performed using a system of Higher Order Logic (HOL) (Hales et al. 2015) whose proof objects (natural-deduction style derivations), whilst tedious, are in principle understandable to a user experienced with both the software and problem domain. The same cannot be said of the Boolean Pythagorean Triples problem, a statement of Ramsey theory involving the

structure of the natural numbers. Rather than taking a high-level approach like HOL, (2016) analysed sets $\{0 \dots n\}$ for larger and larger n , encoding these restricted versions of the problem into the language of boolean satisfiability (SAT), and found that the problem is unsatisfiable for $n = 7825$, and hence for the natural numbers as a whole. In this case, the proof object demonstrates this unsatisfiability using 200 terabytes of propositional logic clauses (compressable to 68 gigabytes). Not only is this far too much for any human to comprehend, but the concepts used in the proof (boolean formulae) are several layers removed from the actual problem statement (natural numbers, subsets and pythagorean triples).

Whilst “low level” formalisms like SAT are less understandable or explanatory for users, they are far more amenable to automation than more expressive logics. Despite the proof for the Boolean Pythagorean Triples problem being many orders of magnitude larger than that of the Kepler Conjecture, the latter is well beyond the ability of today’s automated theorem provers due to its encoding in HOL. Instead, it took 22 collaborators 9 years just to formalise the proof (Hales had previously produced a less formal proof, hundreds of pages long and accompanied by unverified software; yet another reminder that human-generated artefacts are not necessarily understandable either).

Forgoing understandability

It may be the case that, given the increase in power, generality and predictiveness that ML approaches give, and the increasing complexity and amount of scientific knowledge, we decide to forgo understandability in science. As a community we would be in a unique position to develop thinking on this, and to answer questions such as whether we should try to replace understandability with something else. We suggest identifying and engaging with stakeholder groups in science and mathematics to ensure that we develop in directions which will be fruitful and useful to society.

Another possible solution to the problems described here would be to forego understandability in the current sense, or rather to change our notion of what *kind* of thing we are aiming to understand. For instance, could a neural network itself be considered to be a scientific discovery, analogous to the discovery of a new plant? It may be that AI systems become objects of study in the same way as the human brain is currently an object of study, with methods and approaches from neural science, psychology, cognitive science and so on employed to understand an AI system and its behaviour and interactions. There is an interesting analogy between ways in which we can “interrogate” a neural network, for instance via generating inputs aligned to deep features (by specifying a deep-level state, then “training” the input to get close), and how we use introspection and analysis to understand human learning. We’re gradually becoming cognitive scientists and psychologists for the robots.⁵ This is already

⁵The term “robotpsychologist” was coined by Isaac Asimov in (Asimov 1950) to describe the study of the personalities and behavior of intelligent machines.

an active research area, with (Jonas and K.P. 2017) offering a cautionary tale. Again, as a community we would be in a position to provide a unique perspective on this, having reflected on what constitutes an artefact and how they might be evaluated as novel or significant discoveries.

Concluding Remarks

Recent developments in other areas of AI – principally machine learning (ML) – have led to astonishingly rapid progress in generative processes. Research in Constructive Machine Learning has led to impressive generative results in both the arts and sciences, including painting, music, poetry, gaming, drug design, and gene design – usually in collaboration with domain experts. Our concern is that the sheer size and combined resources of the ML community may render generative work in CC untenable, potentially leading to an identity crisis in the field.

CC has long been seen as more than “mere generation”.⁶ Celebrating and automating other aspects of creative acts in addition to generation – such as making aesthetic judgements, producing framing information (background information about the work) and finding new meta-level processes – is partly what distinguishes us from other AI fields. As generative results in neighbouring areas of AI become more sophisticated, we may wish to focus on these other aspects of the creative act. Extending our repertoire to include more scientific domains will further strengthen our communal identity and enhance our value to other AI researchers and to domain experts in science.

There is also the question of whether CC output might run up against natural boundaries in some areas of the arts. For instance, it is possible that in highly expressive domains, such as poetry, computationally produced poems will not be taken seriously, given the lack of authenticity of life experiences they have. This was discussed in (Colton, Pease, and Saunders 2018), in which the authors argue that a lack of authenticity is a looming issue in CC. Authenticity is not so inherently valuable in the sciences.

Furthermore, it *may* be the case that as the novelty and backstory of computer-generated art wears off, society questions whether we want more computer-produced paintings or poems. The question as to whether we still want more computer-generated science or mathematics, however, seems less likely to be asked: we *always* want more science and mathematics. We suggest this only hesitantly. At the turn of the 20th century, photography caused an explosion in the productivity of art. Flooding the market with images forced artists to redefine their value and led to the creation of modern art, transforming individual self-expression. Subareas of photography have themselves developed as unique art forms, such as wildlife photography and photojournalism. Art has further been transformed through digital technology by filters and editing. We can take inspiration from this: advances in AI can saturate old ways of thinking, but naturally open up new ones. If high quality computationally produced

⁶The informal, tongue in cheek slogan at the 2012 ICCA conference was “scoffing at mere generation for more than a decade.” – although this has been challenged, for instance by (Ventura 2016).

art becomes common-place, art as we know it will be transformed forever: a lot of concepts in art might collapse, but at the same time new concepts which are currently unpredictable might emerge. Of course, applying CC to science may equally saturate certain fields or kinds of work. We raise this here to begin a conversation on where CC in the arts may eventually lead, and as a further potential concern about focusing all our energy on the arts.

People are not naturally good at science. The history of science and scientific methodology, the length of time it takes to train a scientist and the high number of published research findings in science which are considered to be false or sub-standard⁷ all hint at the difficulty of the scientific enterprise. This is partially due to political and institutional factors such as pressure to publish, conflicts of interest and a culture which is often more competitive than collaborative; but also partially due to the constant battle to avoid the large number of cognitive biases that adversely affect our reasoning and judgements (Haselton, Nettle, and Andrews 2005; Sutherland 2013). On the other hand, the arts – while also difficult to do well – do not usually go against our natural way of thinking, and can be seen as a celebration of our humanity. In many ways science should be an obvious application domain for computational creativity. This paper is a call to arms for the whole CC community: to apply systems based on cognitive mechanisms to scientific and mathematical domains; to employ experience in building and evaluating interactive systems to this context; and to use expertise in automatically producing framing functionality to enhance the communicability of automatically generated scientific knowledge.

Acknowledgements

We are very grateful to the thoughtful reviews and meta-review. We gratefully acknowledge financial support from EPSRC Grant EP/P017320/1. The fourth author is supported by the NERC National Productivity Investment Fund (NPIF) grant, ref. no. NE/R01051X/1 and the James Watt Scholarship scheme at Heriot-Watt University. The eighth author is supported by the Royal Academy of Engineering under the Research Fellowship scheme.

References

A., K.; Toivanen, J. M.; Xiao, P.; and Toivonen, H. 2014. From isolation to involvement: Adapting machine creativity software to support human-computer co-creation. In *Proc. of the 5th ICCS*.

Asimov, I. 1950. *I, Robot*. Gnome Press.

Baker, M. 2016. 1,500 scientists lift the lid on reproducibility. *Nature* (533):452–454.

Ball, P. 2012. Proof claimed for deep connection between primes. *Nature* 489(7414).

Barendregt, H., and Wiedijk, F. 2005. The challenge of computer mathematics. *Philosophical Transactions of the Royal*

⁷Meta-scientific studies suggest that 85% of biomedical research efforts are wasted (Macleod et al. 2014) and 90% of respondents to a recent survey in Nature agreed that there is a ‘reproducibility crisis’ (Baker 2016) (see (Munafò et al. 2017; Ioannidis 2005) for further details).

Society A: Mathematical, Physical and Engineering Sciences 363(1835):2351–2375.

Barras, B.; Boutin, S.; Cornes, C.; Courant, J.; Filliatre, J.-C.; Gimenez, E.; Herbelin, H.; Huet, G.; Munoz, C.; Murthy, C.; et al. 1997. *The Coq proof assistant reference manual: Version 6.1*. Ph.D. Dissertation, INRIA.

Barton, B. 2009. *The Language of Mathematics: Telling Mathematical Tales*. Mathematics Education Library, Vol. 46. Springer.

Baydin, A. G.; Lopez De Mantaras, R.; and Ontanon, S. 2012. Automated generation of cross-domain analogies via evolutionary computation. In *Proc. of the 3rd ICCS*.

Bond, C. E.; Gibbs, A.; Shipton, Z.; and Jones, S. 2007. What do you think this is? “Conceptual uncertainty” in geoscience interpretation. *GSA Today* 17(11):4–10.

Bray, L., and Bown, O. 2016. Applying core interaction design principles to computational creativity. In *Proc. of the 7th ICCS*.

Bundy, A. 2011. Automated theorem provers: a practical tool? *Ann Math Artif Intell* 61:3–14.

Charnley, J.; Pease, A.; and Colton, S. 2012. On the notion of framing in computational creativity. In *Proc. of the 3rd ICCS*.

Colton, S.; Goodwin, J.; and Veale, T. 2012. Full-face poetry generation. In *Proc. of the 3rd ICCS*.

Colton, S.; Pease, A.; and Charnley, J. 2011. Computational creativity theory: The FACE and IDEA descriptive models. In *Proc. of the 2nd ICCS*.

Colton, S.; Pease, A.; and Saunders, R. 2018. Issues of authenticity in autonomously creative systems. In *Proc. of the 9th ICCS*.

Crevier, D. 1993. *AI: The Tumultuous Search for Artificial Intelligence*. New York, NY: BasicBooks.

Derrida, J. 1987. *The Truth in Painting*. Univ. of Chicago Press.

Dibbets, J. 2002. Interactions between science and art. *Cardiovascular Research* 56(3):330–331.

Došilović, F. K.; Brčić, M.; and Hlupić, N. 2018. Explainable artificial intelligence: A survey. In *41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*, 0210–0215. IEEE.

Džeroski, S.; Langley, P.; and Todorovski, L. 2007. Computational discovery of scientific knowledge. *Computational Discovery: LNAI* 4660:1–14.

Gaff, J. G., and Ratcliff, J. L. E. 1996. *Handbook of the Undergraduate Curriculum A Comprehensive Guide to Purposes, Structures, Practices, and Change - The Jossey-Bass Higher and Adult Education Series*. Wiley.

Gallie, W. B. 1955 – 1956. Essentially contested concepts. *Proceedings of the Aristotelian Society* 56:167–198.

Gonthier, G. 2013. Engineering mathematics: the odd order theorem proof. *POPL* Roberto Giacobazzi and Radhia Cousot (eds):pp. 1–2.

Gowers, T., and Nielsen, M. 2009. Massively collaborative mathematics. *Nature* 461(7266):879–881.

Gowers, T. 2000. Rough structure and classification. *GAGA (Geometric And Functional Analysis)* Special volume – GAGA2000.

Grace, K.; Maher, M. L.; Mohseni, M.; and Perez Y Perez, R. 2017a. Encouraging p-creative behaviour with computational curiosity. In *Proc. of the 8th ICCS*.

Grace, K.; Maher, M.; Mohseni, M.; and Pérez y Pérez, R. 2017b. Encouraging p-creative behaviour with computational curiosity. In *Proc. of the 8th ICCS*.

- Hales, T. C.; Harrison, J.; McLaughlin, S.; Nipkow, T.; Obua, S.; and Zumkeller, R. 2010. A revision of the proof of the Kepler conjecture. *Discrete & Computational Geometry* 44(1):1–34.
- Hales, T.; Adams, M.; Bauer, G.; Dang, D. T.; Harrison, J.; Hoang, T. L.; Kaliszky, C.; Magron, V.; McLaughlin, S.; Nguyen, T. T.; et al. 2015. A formal proof of the Kepler conjecture. *arXiv preprint arXiv:1501.02155*.
- Haselton, M. G.; Nettle, D.; and Andrews, P. W. 2005. The evolution of cognitive bias. In Buss, D. M., ed., *The Handbook of Evolutionary Psychology*. Hoboken, NJ, US: John Wiley and Sons Inc. 724–746.
- Hersh, R. 1991. Mathematics has a front and a back. *Synthese* 88(2):127–133.
- Heule, M. J.; Kullmann, O.; and Marek, V. W. 2016. Solving and verifying the boolean pythagorean triples problem via cube-and-conquer. In *International Conference on Theory and Applications of Satisfiability Testing*, 228–245. Springer.
- Inglis, M., and Aberdein, A. 2015. Beauty is not simplicity: An analysis of mathematicians’ proof appraisals. *Philosophia Mathematica* 23(1):87–109.
- Inglis, M.; Pablo, J.; Mejia-Ramos; Weber, K.; and Alcock, L. 2013. On mathematicians’ different standards when evaluating elementary proofs. *Topics in Cognitive Science* 5(2):270–282.
- Ioannidis, J. P. A. 2005. Why most published research findings are false. *PLOS Medicine* 2(8).
- Jonas, E., and K.P., K. 2017. Could a neuroscientist understand a microprocessor? *PLoS Comput Biol* 13(1).
- Jones, C. B. 2003. The early search for tractable ways of reasoning about programs. *IEEE Annals of the History of Computing* 25(2):26–49.
- Jordanous, A. 2014. Stepping back to progress forwards: Setting standards for meta-evaluation of computational creativity. In *Proc. of the 5th ICC*.
- Juršič, M.; Cestnik, B.; Urbančič, T.; and Lavrač, N. 2012. Cross-domain literature mining: Finding bridging concepts with cross-bee. In *Proc. of the 3rd ICC*, pp. 33–40.
- Karimi, P.; Grace, K.; Maher, M.; and Davis, N. 2018. Evaluating creativity in computational co-creative systems. In *Proc. of the 9th ICC*.
- Krishnan, A. 2009. What are academic disciplines? Some observations on the disciplinarity vs. interdisciplinarity debate. Technical Report 03/09, National Centre for Research Methods.
- Lakatos, I. 1976. *Proofs and Refutations*. Cambridge, UK: CUP.
- Lakoff, G., and Núñez, R. 2000. *Where Mathematics Comes From: How the Embodied Mind Brings Mathematics into Being*. New York: Basic Books.
- Langley, P. 1979. Rediscovering physics with bacon. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pp. 505–507. Tokyo: Morgan Kaufmann.
- Langley, P. 2000. The computational support of scientific discovery. *International Journal of Human-Computer Studies* 53:1149–1164.
- Langley, P. 2002. Lessons for the computational discovery of scientific knowledge. In *Proceedings of First International Workshop on Data Mining Lessons Learned*, 9–12. Sydney: University of New South Wales.
- Li, B.; Zook, A.; Davis, N.; and Riedl, M. 2012. Goal-driven conceptual blending: A computational approach for creativity. In *Proc. of the 3rd ICC*.
- Lindsay, R. K.; Buchanan, B. G.; Feigenbaum, E. A.; and Lederberg, J. 1993. Dendral: A case study of the first expert system for scientific hypothesis formation. *Artificial Intelligence* 61(2):209–261.
- Loughran, R., and O’Neill, M. 2017. Application domains considered in computational creativity. In *Proc. of the 8th ICC*, 197–204.
- Macleod, M. R.; Michie, S.; Roberts, I.; Dirnagl, U.; Chalmers, I.; Ioannidis, J.; Al-Shahi Salman, R.; Chan, A.; and Glasziou, P. 2014. Biomedical research: increasing value, reducing waste. *Lancet* (383):101–104.
- Martin, U., and Pease, A. 2013. Mathematical practice, crowdsourcing, and social machines. In Carette, J.; Aspinnall, D.; Lange, C.; Sojka, P.; and Windsteiger, W., eds., *Intelligent Computer Mathematics*, volume 7961 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. 98–119.
- Munafò, M. R.; Nosek, B. A.; Bishop, D. V. M.; Button, K. S.; Chambers, C. D.; Percie du Sert, N.; Simonsohn, U.; Wagenmakers, E.-J.; Ware, J. J.; and Ioannidis, J. P. A. 2017. A manifesto for reproducible science. *Nature Human Behaviour* 1(1):0021+.
- Newton, R. G. 2000. *The Truth of Science: Physical Theories and Reality*. Cambridge, Massachusetts: Harvard University Press.
- Nielsen, M. 2011. *Reinventing Discovery: The New Era of Networked Science*. USA: Princeton University Press.
- Nipkow, T.; Paulson, L. C.; and Wenzel, M. 2002. *Isabelle/HOL: a proof assistant for higher-order logic*, volume 2283. Springer Science & Business Media.
- O’Donoghue, D., and Keane, M. T. 2012. A creative analogy machine: Results and challenges. In *Proc. of the 3rd ICC*.
- O’Donoghue, D. P.; Abgaz, Y.; Hurley, D.; Ronzano, F.; and Saggion, H. 2015. Stimulating and simulating creativity with dr invento. In *Proc. of the 6th ICC*, pp. 220–227.
- Polya, G. 1945. *How to solve it*. Princeton University Press.
- Polymath, D. 2014. Variants of the selberg sieve, and bounded intervals containing many primes. *Research in the Mathematical Sciences* 1(12).
- Rankey, E. C. 2003. Interpreter’s corner – that’s why it’s called interpretation: Impact of horizon uncertainty on seismic attribute analysis. *The Leading Edge* 22(9).
- Simon, H. A. 1966. Scientific discovery and the psychology of problem solving. In (Ed.), R. C. C., ed., *Mind and cosmos: Essays in contemporary science and philosophy*. University of Pittsburgh Press.
- Sutcliffe, G., and Suttner, C. 2001. Evaluating general purpose automated theorem proving systems. *Artificial intelligence* 131(1):39–54.
- Sutherland, S. 2013. *Irrationality: The enemy within*. Pinter and Martin Ltd.
- Tomašič, P.; Žnidaršič, M.; and Papa, G. 2014. Arts, news, and poetry – the art of framing. In *Proc. of the 5th ICC*.
- Veale, T. 2012. From conceptual “mash-ups” to “bad-ass” blends: A robust computational model of conceptual blending. In *Proc. of the 3rd ICC*, pp. 1–8.
- Ventura, D. 2016. Mere generation: Essential barometer or dated concept? In *Proc. of the 7th ICC*.
- Wason, P. C., and Shapiro, D. 1971. Natural and contrived experience in a reasoning problem. *Quarterly Journal of Experimental Psychology* 23:63–71.

No Time Like the Present: Methods for Generating Colourful and Factual Multilingual News Headlines

Khalid Alnajjar

khalid.alnajjar@helsinki.fi

Leo Leppänen

leo.leppanen@helsinki.fi

Hannu Toivonen

hannu.toivonen@helsinki.fi

Department of Computer Science and HIIT, University of Helsinki, Finland

Abstract

News headlines are the main method for briefly providing a summary of the news article and attracting an audience. In this paper, we experiment with different existing methods for producing colourful expressions and news headlines computationally, in a practical setting. Our case study is conducted by modifying an automated journalism system that generates multilingual news in three languages, namely English, Finnish and Swedish. We adapt existing methods for creative headlines and figurative language generation into the headline generation process of the system, modifying them to work in a multilingual setting. We conduct our evaluation by asking online judges to assess the original titles produced by the unmodified system and those enhanced by the methods described in this paper. The results of the evaluation suggest that the presented methods increase the creativity of existing headlines while maintaining their descriptiveness.

Introduction

The interest in automated journalism has increased in the past years, driven by the ability to produce tailored stories cost-effectively even for small audiences, i.e., the so-called long tail effect. Current methods for automated news generation typically utilize linguistic templates written by journalists, and fill them in with appropriate information from structured data sources. Template-based methods give strong control over the output generated by the system and ensures conveying the message as intended. However, news produced by these approaches tend to be repetitive and sound mechanistic.

Headlines are an essential part of the news. They must relate to the news article and briefly describe it while motivating readers to visit and read the article. Automated journalism systems aim to produce informative headlines, but not colourful ones requiring creativity.

In this paper, we experiment with different existing methods for creating colourful expressions and with their use in template-based news headlines. We seek for a balance between creativity and factuality. Because of the latter, we build on an existing template-based system that produces factual headlines; for the former, we generate creative expressions and add them to the factual headlines.

For our case study, we use a modified version of *Valtteri*¹ (Leppänen et al. 2017), an automated journalism system, as the baseline. *Valtteri* generates election news about the 2017 Finnish municipal election results in three languages, English, Finnish and Swedish. For the scope of this work, we focus on two languages only: English and Finnish.

Creativity, such as use of figurative language, is something human journalists consider to be one of their strengths when compared to automated journalism systems (van Dalen 2012). Creativity is missing from most, if not all, automated journalism systems is creativity. This also applies to *Valtteri*.

Inspired by previous research on generating figurative language (Veale and Li 2013; Alnajjar et al. 2017) and creative headlines (Lynch 2015; Gatti et al. 2015), we present two methods which add a creative touch to news headlines generated by the automated journalism system. The methods are developed to operate in a multilingual setting. The first method finds a suitable well-known phrase (e.g. movie title) to be presented to the reader as a catchy title (i.e. it draws attention) along with the factual message. The other method injects figurative phrases (e.g. similes and metaphors) into headlines, depending on the polarity of the news. We exploit recent research in word cross-lingual embeddings, permitting us to project knowledge from English, with rich linguistic resources, into a less-resourced one, i.e. Finnish.

In our evaluation, we crowdsourced the assessment of the headlines to online judges (acting as the audience). We asked them to evaluate the original headline produced by *Valtteri* and the new altered headlines produced by the methods described in this paper in order to test the applicability of these methods in a practical scenario. The judges were asked to assess aspects such as informativeness, correctness and catchiness, to measure the effects of figurative modifications on the original headlines. Because of the availability of crowdsourcing workers, the current evaluation is conducted on English headlines only and Finnish is left for future work.

This paper is structured as follows. We begin by reviewing related work on headline generation. Thereafter, we describe the *Valtteri* system and how the creative component is attached to the system. We then elucidate the methods employed by us to convert the headlines generated by *Valtteri*

¹ <https://www.vaalibotti.fi/>

into more colourful ones. The evaluation details are then provided, followed by the results. Lastly, we discuss the results and conclude this work.

Related Work

Previous research on headline generation is extensive, covering different approaches based e.g. on rules, statistics, summarization or machine learning. In this section, we briefly describe the most relevant work.

Hedge Trimmer (Dorr, Zajic, and Schwartz 2003), a rule-based method for headline generation, decides which words are to be retained and which to be pruned from the news article. Their rules are linguistically motivated and based on analyzing human-made headlines written in English. Building such rules is tedious, especially when dealing with multilingual news articles. Wang, Dunnion, and Carthy (2005) extended the work by introducing a C5.0 decision tree classifier for predicting which words to include in the title.

Zajic, Dorr, and Schwartz (2002) use a Hidden Markov Model to generate news headlines for a news story by having the model capture keywords from the beginning, i.e. first paragraphs, of the story. A Viterbi Decoding algorithm is then applied to headlines generated by the model to find the most representative headline. Additionally, four decoding parameters are imposed to ensure the quality of the generated headline, namely: (1) a length penalty, to keep headlines within the 5 to 15 word length limits, (2) a position penalty, to give a higher penalty to words appearing later in the story, (3) a string penalty, to encourage neighbouring words and (4) a gap penalty, to reduce the distance between selected words. Another statistical approach (Colmenares et al. 2015) uses sequence prediction methods for learning how humans craft headlines. Given a story, their model classifies whether a certain token in the story should be in the headline or not. In the case of a token being classified as in-headline, their method considers various features regarding the text of the story, the token (e.g. parts-of-speech tags and name-entities) and the constructed headline at each stage. Other statistical-based research on headline generation has been conducted by Banko, Mittal, and Witbrock (2000), Knight and Marcu (2002), Wan et al. (2003) and Unno et al. (2006).

Summarization-based techniques treat the problem of headline generation as producing a one sentence digest of the article (Morita et al. 2013; Martins and Smith 2009; Filippova 2010). Summarization techniques tend to extract and then compress sentences existing in the new article, which results in reusing words/phrases existing in the article. Furthermore, deep learning models have also been employed in the generation of headlines by learning how to summarize a certain text (Ayana et al. 2016). Such models require sufficiently big training data sets which can be prohibitively large for some scenarios.

A way of expressing headlines in various styles is to learn different ways of talking about the same news article. Wubben et al. (2009) have proposed a way of grouping news articles from different sources based on the content similarity. Using the different ways of writing a headline for a certain topic, a machine translation model could be trained to learn how to paraphrase headlines (Wubben, Bosch, and

Krahmer 2010). *HEADY* (Alfonseca, Pighin, and Garrido 2013), on the other hand, performs event pattern clustering and generates a headline for an unseen news article by inferring headlines based on the events in it.

The above approaches do not consider an important aspect of news headlines, which is catchiness. To our best knowledge, catchiness in news headlines generation is addressed only in the work by Lynch (2015) and Gatti et al. (2015).

Lynch (2015) proposed a system for adding a well-known phrase (e.g. songs, films ... etc) as a prefix to an existing title. The added phrase is intended to catch the attention of the readers and increase search engine optimization. The system extracts keywords from an article, clusters and expands them. Then, it pairs keywords from distinct clusters if they co-occurred in a corpus of 5-grams. Using a pseudo-phonetic string matching algorithm and semantic similarity measurement, the system finds and ranks well-known phrases suitable for the pair. Lastly, it embeds the matched well-known phrase in the existing headline.

In the method described by Gatti et al. (2015), titles are given a creative touch by blending them with well-known expressions. Their headline generation process extracts keywords from the input news article. Thereafter, the method finds existing well-known phrases that are semantically similar to the existing headline and the article. These phrases are then modified by altering a word in them that satisfies a semantic similarity threshold, and lexical and syntactic constraints.

Despite the advances in automated headline generation, research on generating catchy and diverse headlines for automated journalism is scarce, especially in a multilingual setting with less-resourced languages.

Adding Creativity to Valtteri Headlines

Valtteri (Leppänen et al. 2017; Melin et al. 2018) is a multilingual system for automated journalism, reporting on the 2017 Finnish municipal elections. The system follows a data-driven approach to generate news while ensuring certain requirements, e.g. accuracy (i.e. factual and not misleading) of the produced news.

We add the creativity component to the system at a central stage of the pipeline, immediately after the aggregation process. It has access to the data and the selected templates to be used in the news. The component can alter the content of the news article produced along with its headline.

Inspired by existing research on computational linguistic creativity and creative headline generation, we implement two methods for producing colourful headlines. The methods are:

1. **Phrase-copying:** We find and insert a suitable well-known phrase into a factual headline (Lynch 2015; Gatti et al. 2015).
2. **Figurative-injection:** We generate figurative expressions using linguistic patterns and knowledge-bases of stereotypical properties of nouns (Veale and Li 2013; Alnajjar et al. 2017), and insert them into existing headlines.

For our use case, these methods should be incorporated in the automated journalism system and they should work

in multiple languages. To achieve this, in case the required linguistic resources are not available for Finnish, we resort to pre-trained and aligned multilingual word embeddings ζ (Bojanowski et al. 2017; Joulin et al. 2018). In these models, a vector representation of a word in a certain language (e.g. *king* in English, ζ_{en}) should roughly point to the same semantic direction in another model (e.g. *kuningas* in Finnish, ζ_{fi}) and vice versa. With the help of these aligned models, we can exploit available linguistic creativity resources in English and project them into Finnish.

The following sub-sections describe the two methods for colourful headline generation in-depth.

Phrase-copying: Insertion of Well-Known Phrases

Inspired by the research by Lynch (2015) and Gatti et al. (2015), we implement a method for finding and inserting well-known phrases into headlines produced by *Valtteri*. The results have the form “*phrase: headline*”, c.f. Table 1 for examples. Juxtapositioning the phrase with the headline is expected to catch the attention of viewers and motivate them to click on the headline to read the news article, while keeping the factual content of the original headline intact. For this to work as intended, the method should find a well-known phrase that matches the original headline.

We use two types of well-known phrases: proverbs and movie titles. Proverbs for each language are extracted from wikiquote.org². Regarding movie titles, we use the dataset of movies provided by IMDB³. We restrict the dataset to movies with more than 100,000 votes, to exclude generally unfamiliar titles. As these titles are in English and we desire to know how they are known to people in other languages, we query *Wikipedia* with the movie title in English and retrieve the title of its corresponding Finnish *Wikipedia* article. As an example, the movie title “Harry Potter and the Philosopher’s Stone” is known to Finns as “Harry Potter ja viisasten kivi”.

We perform a preprocessing step on the collected phrases to clean and expand them. The process commences by stripping punctuation and any parentheses including the content in them to omit some explanations given in the proverbs. We also removed phrases containing more than 5 words to avoid lengthy headlines that could distract the audience. Some movie titles separate a general title and a subtitle by a colon or a dash; we include in our dataset both the short version (before the colon/dash) and the long version (all of the text).

In total, the database of well-known phrases contains 1,744 and 1,322 phrases in English and Finnish, respectively. We denote this database by P .

In order to identify a well-known phrase that matches the headline, two aspects are checked: 1) semantic similarity (or relatedness) between the phrase and the headline, and 2) prosody of the phrase and the headline. Semantic similarity is used for coherence of the resulting combination, while

prosody is evaluated to increase catchiness of the result.

We employ a greedy algorithm to match phrases to a given headline H . For each phrase ρ in P , the method computes the cosine semantic similarity between individual words w_1, w_2 in ρ and H , using the corresponding language model ζ_l , where l is either ‘en’ or ‘fi’, as follows:

$$sim_{words}(w_1, w_2, t, l) = \begin{cases} \zeta_l(w_1, w_2), & \text{if } \zeta_l(w_1, w_2) \geq t \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

$$sim_{phrases}(H, \rho, t, l) = \sum_{w \in H} \sum_{k \in \rho} sim_{words}(w, k, t, l) \quad (2)$$

In equations 1 and 2, t is a threshold for the minimum semantic similarity desired. We empirically set t to 0.3. While increasing the threshold t would find phrases that are more semantically similar to the headline, it would also narrow the space of possible solutions, especially for languages other than English. If a phrase has received a semantic similarity score on Equation 2 greater than 0, it is considered to be similar to the headline.

When a phrase ρ is found to be semantically similar to headline H , the method computes four prosody features between the matched phrase and the headline. These features are assonance, consonance, alliteration and rhyme. We utilize the *espeak-ng tool*⁴ to acquire the International Phonetic Alphabet (IPA) transcriptions of words. The tool supports producing IPA for multiple languages, including English and Finnish. For each word pair in $\rho \times H$, the method evaluates whether the pair has phonetic similarity of any of the four prosody features. Then, each prosody feature is aggregated to phrase-level by computing its average score over the word pairs in $\rho \times H$.

Finally, we aggregate all four features into one number by obtaining a weighted sum of their phrase-level scores. We assigned to rhyme and alliteration weights of 40% each, and to consonance and assonance weights of 10% each, based on empirical testing.

Given a headline H , the method considers phrases ρ in P in a random order, computing the above measures of semantic similarity and prosody. The method keeps progressing until it finds ten well-known phrases that have a positive semantic similarity and a positive phonetical similarity with headline H . Among the ten phrases, the method then picks the phrase with the highest prosody score. The magnitude of the semantic similarity is not considered, since the template-based headline generation method tends to give the highest semantic similarity to the same phrases; for prosody, there is more variation based on how the template has been instantiated.

Finally, the selected phrase is inserted into the headline. For headline examples generated by this method, see Table 1.

²English: [https://en.wikiquote.org/wiki/English_proverbs_\(alphabetically_by_proverb\)](https://en.wikiquote.org/wiki/English_proverbs_(alphabetically_by_proverb))
Finnish: https://en.wikiquote.org/wiki/Finnish_proverbs

³<https://datasets.imdbws.com/>

⁴ <https://github.com/espeak-ng/espeak-ng>

#	Baseline	Phrase-copying	Figurative-injection
(1)	Most seats go to The Centre Party of Finland in Kangasniemi	Legends of the Fall: Most seats go to The Centre Party of Finland in Kangasniemi	Most seats go to The Centre Party of Finland, the free queen, in Kangasniemi
(2)	Biggest vote gains for The Green League in Kuopio	Alls well that ends well: Biggest vote gains for The Green League in Kuopio	Biggest vote gains for The Green League –the lovely god– in Kuopio
(3)	Biggest gains for The Christian Democrats across Lapin vaalipiiri	The Running Man: Biggest gains for The Christian Democrats across Lapin vaalipiiri	Biggest gains for The Christian Democrats, as powerful as a soldier, across Lapin vaalipiiri
(4)	Second largest gains for The Christian Democrats in Rovaniemi	The Transporter: Second largest gains for The Christian Democrats in Rovaniemi	Second largest gains for The Christian Democrats –the king– in Rovaniemi
(5)	The Finns Party lose three seats in Jyväskylä	To each his own: The Finns Party lose three seats in Jyväskylä	Like a spy, The Finns Party lose three seats in Jyväskylä

Table 1: Five examples of generated headlines from an existing headline by the two presented methods in this paper, in English.

Figurative-injection: Generation of Figurative Language

The figurative-injection method inserts figurative language (e.g. metaphors and similes) into existing headlines. See the column ‘Figurative-injection’ of Table 1 for examples of headlines generated by this method. We next describe the method.

If the given headline has polarity with respect to the main entity in the headline, a political party or candidate in our case, then the method adds a figurative comparison to an adjective and common noun that is stereotypically associated with the polarity. The aim is that this comparison indirectly attributes properties to the entity of the headline, thereby emphasizing the polarity in a creative, figurative way.

Given that the automated journalism system works with structured data and given templates, we can directly associate polarities with the templates and values used to populate them, and avoid the need for automated polarity analysis of headlines. The polarity is determined by inspecting the reported result (i.e. the gains or losses of votes and seats) in the headline, while taking negations into account. In the cases where the headline states that an entity has received a positive result (e.g. majority of votes, biggest gains ... etc) or negative result (e.g. no seats, lose X seats ... etc) it is classified accordingly; otherwise, it is considered to be neutral. Neutral headlines are not modified by this method.

Identification of suitable adjectives and common nouns proceeds in three steps, performed once as a pre-processing step. First, we have manually listed seed nouns that match the election domain (e.g. win, success; loss, defeat). Second, we use corpus-based methods to identify adjectives associated to the seed nouns (e.g. heroic; tragic). Third, we identify common nouns that are stereotypically associated to these adjectives, using an existing knowledge base. We next detail these steps.

First, we manually define a set of seed nouns describing each of the polarities:

- *positive*: win, gain, accomplishment, success, achievement
- *negative*: loss, defeat, failure

Second, using the seed words, we mine stereotypical properties related to them. We observe trigrams in Google N-Grams (Brants and Franz 2006) that match the linguistic pattern “a/n * SEED”, where SEED is any of the seed words, as conducted in previous research by Veale and Li (2013). We retrieve the adjectival properties that occur at the wildcard position (“*”) in such trigrams. We then use the resource by Alnajjar et al. (2017) to prune out noisy and non-adjectival relations (e.g. “a 3-5 win”). Examples of mined properties for the two categories are: “a *heroic* achievement” and “a *tragic* loss”.

Some positive adjectives can be associated with negative situations (e.g. “a *great* loss”). We use the polarity function provided in *Pattern* library (De Smedt and Daelemans 2012) to predict the polarity of adjectives, and we filter out any adjectival property that has a polarity which does not match the intended classification.

Third, the method looks for suitable metaphorical nouns (common nouns in our case) that are strongly associated with the desired properties. For this, we use a tested dataset κ of nouns and their weighted stereotypical properties (Alnajjar et al. 2017). An example of a noun and its stereotypical properties along with their weights is *King*: {powerful: 1563, successful: 1361, ... etc}.

Given a headline to modify, the method now has access to knowledge of which properties describe a positive or negative situation and which nouns are well-known to possess these properties. The method then searches for a suitable metaphorical noun to be introduced in the headline. It does so by iterating over all the properties describing the situation and the common nouns in κ to find out which nouns are associated to many of these properties. In the process, the method keeps track of all these nouns and how strongly they are related to the relevant properties in knowledge-

base κ . Thereafter, the nouns are sorted based on the sum of their association weights. A random noun having a total weight above the third quartile of weights is selected to be the metaphorical noun. A random stereotypical property of the selected noun is then chosen while ensuring that it meets two constraints: 1) it is strongly associated with the noun (i.e. in the top 50%) and 2) it describes the situation. The selected noun and its property will be used, in the remainder of this method, to construct a figurative expression.

The knowledge-base κ and the linguistic pattern used to find adjectival properties are in English but we desire to generate figurative language in multiple languages. To overcome this obstacle, we employ aligned word embedding models between multiple languages (English and Finnish) as follows. When the method is requested to generate a figurative expression for a language other than English, it begins by using the trigrams and knowledge available in English to find suitable a suitable noun and property. Once a noun and a property are selected, the method obtains their vector representations in the English model. These vectors are then projected into the other aligned model (i.e. Finnish). We consider the closest word to the projected vector as the representation of the word in the other language.

To realize a figurative expression using the selected metaphorical noun and property, we hand-crafted a set of figurative templates in both languages, given in Table 2. For each template, we define whether the template should be injected in the headline before or after the name of the entity. Depending on the position of the entity’s name in the headline, a random figurative template is chosen.

English	Finnish	Position
, as <i>PROPERTY</i> as [a\n] <i>NOUN</i> ,	, <i>PROPERTY</i> kuin <i>NOUN</i> ,	after
, the <i>NOUN</i> ,	, <i>NOUN</i> ,	after
–the <i>NOUN</i> –	– <i>NOUN</i> –	after
, the <i>PROPERTY NOUN</i> ,	, <i>PROPERTY NOUN</i> ,	after
–the <i>PROPERTY NOUN</i> –	– <i>PROPERTY NOUN</i> –	after
Like [a\n] <i>NOUN</i> ,	Kuin <i>NOUN</i> konsanaan,	before
Like [a\n] <i>PROPERTY NOUN</i> ,	Kuin <i>PROPERTY NOUN</i> ,	before

Table 2: Hand-crafted figurative templates in English and Finnish to be injected in existing headlines. The position column indicates whether the template should be injected before or after the entity name.

Finally, the chosen template gets filled with the selected noun and property. To ensure producing grammatically correct metaphorical expressions, we use *Pattern* to reference nouns and properties correctly, for English. Regarding Finnish, we analyze and inflect the projected words in the Finnish space into the nominative form, if necessary, using *UralicNLP* (Hämäläinen 2019) and *Omorfi* (Pirinen 2015).

Evaluation

We asked online judges on *figure-eight.com* to evaluate both the baseline (non-creative) headlines produced by *Valtteri* and the modified (creative) headlines by the methods

described above. As Finnish is not supported by the crowdsourcing platform, we only evaluated English headlines at this stage.

Our evaluation dataset is constructed as follows. We randomly selected a pair of a location and an entity in Finland and passed them to *Valtteri* to obtain the news article covering the election results of the entity in that location, in English. For locations, we only considered the ones on country, district or municipality levels, to exclude news for small areas. In case the reported news by *Valtteri* was classified to be neutral in its polarity, then another random pair was selected. This process was repeated until we had 100 news articles.

The headline of each generated news article was then passed to the creativity component, which generated two modified headlines using the two presented methods. Table 1 shows examples of headlines generated by the methods.

Overall, the evaluation dataset contains 300 English headlines: 100 from the baseline system and 100 generated by both methods. We asked 10 online judges to evaluate each headline. Judges were given a brief description of the task, and the first paragraph of the news story generated by *Valtteri*. They were then asked to evaluate the headline on a 5-point Likert scale against the following claims:

1. The headline is descriptive of the article.
2. The headline is grammatically correct.
3. The headline is catchy.
4. The headline is creative.
5. The headline can be considered offensive.
6. The headline is generated by a computer.

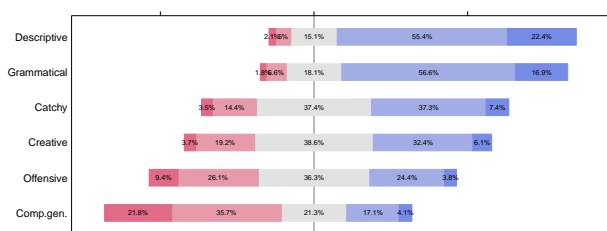
Some of these perspectives are from the prior research by Lynch (2015) and they should be self-explaining.

The quality control mechanism enforced in crowdsourcing was that a minimum of 10 seconds was spent in answering questions about five headlines, in order to eliminate spammers that answer them randomly. We did not apply other measures since the questions and interpretations are subjective and do not have correct answers.

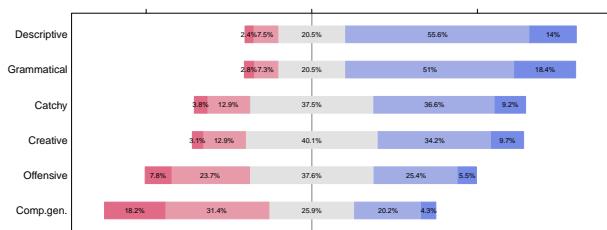
Results

The evaluation process resulted in 3,000 unique judgments from crowdsourcing, 1,000 for each type of headlines. Table 3 shows the mean and standard deviation of judgments received on each question for the three types of headlines, and Figure 1 gives the diverging bar charts for the answers. We next look at the results for each property assessed.

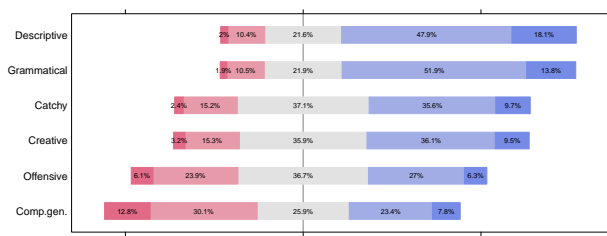
Descriptive From the results, it appears that the three types of generated headlines were considered to be descriptive on average (i.e. $\mu_x > 3$). Despite all versions of the headline having the same factual message present, the headlines produced by *Valtteri* (the baseline) were judged to be the most descriptive. This difference is statistically significant.



(a) Baseline



(b) Phrase-copying



(c) Figurative-injection

■ Strongly Disagree ■ Neutral
■ Agree ■ Strongly Agree

Figure 1: Diverging bar charts illustrating the percentage of judgments received on each question for the three types of headlines.

Grammatical The results concerning the grammaticality of produced headlines is similar to the results on their descriptiveness. That is, all methods produced grammatically correct headlines on average, with headlines produced by the baseline being statistically significantly the most grammatically correct.

Catchy In terms of the catchiness of the headlines, both non-baseline methods have slightly improved the catchiness of original headlines. The difference, however, is not statistically significant.

Creative Regarding the creativity of headlines, the phrase-copying method is perceived to be the most creative, on average. The judges deemed both non-baseline methods to be more creative than the baseline (with statistical significance), as both methods have increased the agreements by approximately 6%.

Offensive Headlines produced by the non-baseline methods are more likely to produce offensive headlines. The difference between the baseline and the proposed methods is statistically significant. However, headlines are generally neutral and not offensive (i.e. $\mu_x \leq 3$).

Generated Headlines produced by the non-baseline methods are considered to be computer-generated more often than the ones generated by the baseline methods, to a statistically significant degree. However, headlines produced by all variations could pass as being written by humans as the majority of judges believed that they are not generated by computers.

Discussion

The aim of the proposed methods was to add creative language to news headlines, in order to add variation to them and to make them more interesting for readers.

According to our empirical results, the proposed methods indeed improved the creativity of the original headlines produced by *Valtteri*. This shows that the methods had some success in making the headlines more creative.

By adding creative elements, we also aimed to make the headlines more catchy. Here the methods were only slightly successful: catchiness was improved marginally. This result shows that creativity does not necessarily improve catchiness in the case of headline generation.

The modified headlines lost some of the descriptiveness of the original headlines, indicating that the added elements did not match the contents of the headline or the news story. In the case of the phrase-copying method, the main problem seems to be that despite our aim to choose phrases that are semantically related to the original headline, the added phrases can still be poorly chosen. Our measure of semantic similarity considers relations between individual words, but does not in any way take into account the meanings or mental images of the phrases as a whole. Adding a phrase with polarity matching the polarity of the headline could help, but more work is needed to make better use of well-known phrases given their rich, cultural meanings and interpretations. For the figurative-injection method, the result suggests that the selection of nouns and adjectives, but also the design of the templates used to inject figurative expressions, should be improved.

The modified headlines also lost some of their grammatical correctness. This is somewhat surprising for the phrase-copying method whose results consist of a well-known phrase and the original headline. Technically speaking, one would expect these to be grammatically about equally correct with the original headlines. A possible explanation is that the decrease in perceived grammatical correctness is influenced by poor matching of the added phrase and the original headline, as discussed above. An alternative cause is that the judges did not recognize all “well-known” phrases and therefore did not see the (grammatical) point in the generated headline. In the case of the figurative-injection method, the result implies again that the templates used to inject figurative expressions should be improved for grammatical fluency.

	Descriptive		Grammatical		Catchy		Creative		Offensive		Comp.gen.	
	μ_x	SD	μ_x	SD	μ_x	SD	μ_x	SD	μ_x	SD	μ_x	SD
<i>Baseline</i>	3.91	0.87	3.80	0.86	3.31	0.93	3.18	0.94	2.46	1.13	2.87	1.01
Phrase-copying	3.75*	0.93	3.71*	0.88	3.35	0.95	3.35*	0.93	2.61*	1.12	2.97*	1.01
Figurative-injection	3.70*	0.95	3.65*	0.91	3.35	0.93	3.33*	0.95	2.83*	1.15	3.04*	1.00

Table 3: The mean μ_x and standard deviation SD of judgments received for each type of generated headlines on the six questions. The best result for each question appears in boldface.

* The value is statistically significantly different ($p < 0.05$) from the value for the baseline headline (non-parametric permutation test with one hundred million repetitions, one-tailed, not corrected for multiple testing).

We also assessed whether the modified headlines are more likely to be offensive than the original headlines. This turned indeed to be the case. By inspecting the headlines which were considered to be the most offensive, we noticed that they were usually negative expressions generated by the figurative-injection method. By construction, the method compares a party or person to a common noun, and therefore negative analogs easily become offensive to the involved party. The two most offensive headlines are 1) “No seats for The Christian Democrats, the thief, in Nousiainen” and 2) “Like a fool, The Finns Party drop most seats in Mynämäki”. This result and examples highlight that care needs to be taken when using automated creativity methods to talk about persons (or parties), in order to avoid unintentional offensive expressions. The proposed methods could be modified to reduce the chances of producing offensive outputs as follows: 1) introduce a dictionary of taboo words to filter out risky words or well-known phrases containing them and 2) use a lower threshold when searching for metaphorical nouns, in order to allow for a wider selection of (safe) words. A better but bigger change would be to produce figurative comparisons to the events in the news, such as loss of seats, rather than to the persons or parties involved. Nevertheless, the final output cannot be guaranteed to be safe for production unless it is verified by a human.

Finally, the modified headlines generated by the proposed methods were recognized to be computer-generated more often than the original (computer-generated) headlines. This suggests that the methods to select and inject materials need to be improved, as the eventual goal is produce headlines that appear less computer-generated than the baseline method.

Conclusion

In this paper, we have presented methods for modifying an existing headline generation method, in order to give the headlines a creative touch. The methods work by inserting well-known phrases or figurative language in the headline templates. In our use case, we extended the headline generation method of *Valtteri*, a system that generates news reports on the 2017 Finnish elections in English, Finnish and Swedish. We also described how the methods can utilize cross-lingual links between Wikipedia articles and aligned multilingual word embedding models in order to take advan-

tage of English resources when producing Finnish headlines, but this aspect was not evaluated due to lack of crowdsourcing workers.

Our empirical evaluation using English headlines generated by the proposed methods shows that they made the headlines more creative, and also slightly more catchy, but at the same time we observed a decrease in how descriptive and grammatically correct the headlines are.

In future work, we plan to improve the methods to select and inject materials to headlines, taking better into account the implied meanings of the added phrases or expressions, as well as making the results linguistically more fluent. Evaluation of Finnish headlines will help assess how well the cross-lingual aspects of the methods work. Interesting topics for future work also include automatic extraction of templates for injection of figurative expressions, and production of apt, yet ethically appropriate, figurative expressions. Finally, it would be interesting to introduce figurative language in the body of automatically generated news, not only headlines.

Acknowledgments

This work has been supported by European Union’s Horizon 2020 research and innovation programme under grant agreement No 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media).

References

- Alfonseca, E.; Pighin, D.; and Garrido, G. 2013. Heady: News headline abstraction through event pattern clustering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, 1243–1253. Sofia, Bulgaria: Association for Computational Linguistics.
- Alnajjar, K.; Hämäläinen, M.; Chen, H.; and Toivonen, H. 2017. Expanding and weighting stereotypical properties of human characters for linguistic creativity. In *Proceedings of the 8th International Conference on Computational Creativity*, 25–32. Atlanta, United States: Georgia Institute of Technology.
- Ayana; Shen, S.; Liu, Z.; and Sun, M. 2016. Neural headline generation with minimum risk training. *CoRR* abs/1604.01904.

- Banko, M.; Mittal, V. O.; and Witbrock, M. J. 2000. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, 318–325. Hong Kong: Association for Computational Linguistics.
- Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5:135–146.
- Brants, T., and Franz, A. 2006. *Web 1T 5-gram Version 1*. Linguistic Data Consortium, Philadelphia, PA. Philadelphia, PA.
- Colmenares, C. A.; Litvak, M.; Mantrach, A.; and Silvestri, F. 2015. Heads: Headline generation as sequence prediction using an abstract feature-rich space. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 133–142. Denver, Colorado: Association for Computational Linguistics.
- De Smedt, T., and Daelemans, W. 2012. Pattern for Python. *Journal of Machine Learning Research* 13:2063–2067.
- Dorr, B.; Zajic, D.; and Schwartz, R. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL 03 Text Summarization Workshop*.
- Filippova, K. 2010. Multi-sentence compression: Finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, 322–330. Beijing, China: Coling 2010 Organizing Committee.
- Gatti, L.; Özbal, G.; Guerini, M.; Stock, O.; and Strapparava, C. 2015. Slogans are not forever: Adapting linguistic expressions to the news. In *Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI'15*, 2452–2458. AAAI Press.
- Hämäläinen, M. 2019. Uralicnlp: An NLP library for Uralic languages. *Journal of Open Source Software* 4(37):1345.
- Joulin, A.; Bojanowski, P.; Mikolov, T.; Jégou, H.; and Grave, E. 2018. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2979–2984. Brussels, Belgium: Association for Computational Linguistics.
- Knight, K., and Marcu, D. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence* 139(1):91 – 107.
- Leppänen, L.; Munezero, M.; Granroth-Wilding, M.; and Toivonen, H. 2017. Data-driven news generation for automated journalism. In *Proceedings of the 10th International Conference on Natural Language Generation*, 188–197. Santiago de Compostela, Spain: Association for Computational Linguistics.
- Lynch, G. 2015. Every word you set: Simulating the cognitive process of linguistic creativity with the PUNdit system. *International Journal of Mind Brain and Cognition* 6(1-1).
- Martins, A. F. T., and Smith, N. A. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, ILP '09, 1–9. Boulder, Colorado: Association for Computational Linguistics.
- Melin, M.; Bäck, A.; Södergård, C.; Munezero, M. D.; Leppänen, L. J.; and Toivonen, H. 2018. No landslide for the human journalist—an empirical study of computer-generated election news in finland. *IEEE Access* 6:43356–43367.
- Morita, H.; Sasano, R.; Takamura, H.; and Okumura, M. 2013. Subtree extractive summarization via submodular maximization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, 1023–1032. Sofia, Bulgaria: Association for Computational Linguistics.
- Pirinen, T. A. 2015. Development and use of computational morphology of Finnish in the open source and open science era: Notes on experiences with omorfi development. *SKY Journal of Linguistics* 28:381–393.
- Unno, Y.; Ninomiya, T.; Miyao, Y.; and Tsujii, J. 2006. Trimming CFG parse trees for sentence compression using machine learning approaches. In *Proceedings of the COLING/ACL on Main Conference Poster Sessions*, COLING-ACL '06, 850–857. Sydney, Australia: Association for Computational Linguistics.
- van Dalen, A. 2012. The algorithms behind the headlines. *Journalism Practice* 6(5-6):648–658.
- Veale, T., and Li, G. 2013. Creating similarity: Lateral thinking for vertical similarity judgments. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, 660–670. Sofia, Bulgaria: Association for Computational Linguistics.
- Wan, S.; Dras, M.; Paris, C.; and Dale, R. 2003. Using thematic information in statistical headline generation. In *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering - Volume 12*, Multi-SumQA '03, 11–20. Sapporo, Japan: Association for Computational Linguistics.
- Wang, R.; Dunnion, J.; and Carthy, J. 2005. Machine learning approach to augmenting news headline generation. In *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*.
- Wubben, S.; van den Bosch, A.; Krahmer, E.; and Marsi, E. 2009. Clustering and matching headlines for automatic paraphrase acquisition. In *Proceedings of the 12th European Workshop on Natural Language Generation*, ENLG '09, 122–125. Athens, Greece: Association for Computational Linguistics.
- Wubben, S.; Bosch, A. v. d.; and Krahmer, E. 2010. Paraphrasing headlines by machine translation: Sentential paraphrase acquisition and generation using google news. *LOT Occasional Series* 16:169–183.
- Zajic, D.; Dorr, B.; and Schwartz, R. 2002. Automatic headline generation for newspaper stories. In *Workshop on Automatic Summarization*, 78–85. Philadelphia, PA, USA: Association for Computational Linguistics.

Modelling the Socialization of Creative Agents in a Master-Apprentice Setting: The Case of Movie Title Puns

Mika Hämäläinen

Department of Digital Humanities
Faculty of Arts
University of Helsinki
mika.hamalainen@helsinki.fi

Khalid Alnajjar

Department of Computer Science
Faculty of Science
University of Helsinki
alnajjar@cs.helsinki.fi

Abstract

This paper presents work on modelling the social psychological aspect of socialization in the case of a computationally creative master-apprentice system. In each master-apprentice pair, the master, a genetic algorithm, is seen as a parent for its apprentice, which is an NMT based sequence-to-sequence model. The effect of different parenting styles on the creative output of each pair is in the focus of this study. This approach brings a novel view point to computational social creativity, which has mainly focused in the past on computationally creative agents being on a socially equal level, whereas our approach studies the phenomenon in the context of a social hierarchy.

Introduction

The master-apprentice approach, as introduced by (Alnajjar and Hämäläinen 2018), to computational creativity has been shown to achieve creative autonomy and its creativity has been thoroughly discussed and motivated. However, the question that has remained without an answer has been the social nature of a master-apprentice pair and its effect on the creative outcome.

The approach consists of two parts: a master, which is a genetic algorithm, and an apprentice, which is an LSTM sequence-to-sequence model. While the master is in charge of the internal appreciation of the overall system as implemented in its fitness function, the apprentice plays a crucial role in the creative autonomy as it can learn its standards partially from its master and partially from its peers.

This paper focuses on the exploration of the master-apprentice approach from a social psychological point of view. By modelling the socialization of the apprentice into a creative society consisting of the master and peers, we seek to gain a deeper understanding of the phenomenon in terms of the overall creativity of the system. In addition, modelling the social aspects of a computationally creative system can help in understanding creativity as a social phenomenon in a broader sense (Saunders and Bown 2015).

We motivate the model of socialization based on research conducted on the field of social psychology, namely developmental psychology. We select the categorization of parenting styles presented by (Baumrind 1991) as the theoretical foundation of our work.

The creative task we are tackling in this paper is the creation of humorous movie titles delivering a food-related pun. This consists of taking an existing movie title such as *Beauty and the Beast* and making a pun out of it such as *Beauty and the Beets*. As people have been writing funny movie titles of this sort in a great abundance on the social media, we can gather parallel data easily.

Related Work

While pun generation has been vastly studied in the field of computational creativity (Ritchie 2005; Yu, Tan, and Wan 2018; He, Peng, and Liang 2019), we see that the most important contribution of our paper lies in the realm of social creativity. Therefore, we dedicate this section in describing some of the practical research conducted in the computational social creativity.

Research on an agent community consisting of self-organizing maps (Honkela and Winter 2003), although outside of the computational creativity paradigm, presents a way of simulating the emergence of language. The agents are capable of meaning negotiation and converging into a common language to communicate about edibility of different food items in their shared world.

Multi-agent systems have been studied in the context of novelty seeking in creative artifact generation (Linkola, Takala, and Toivonen 2016). In their study, the agents exert self-criticism and they can vote and veto on creative artifacts. Their findings suggest that multiple creative agents can reach to a higher number of novelty in their output than a single agent system.

A recent study (Hantula and Linkola 2018) has been conducted in social creativity in agent societies where the individuals are goal-aware. The individuals create artifacts of their own and peer up to collaborate with another agent. The agents are capable of learning a peer model that guides them in selecting a collaboration partner.

The papers discussed in this section, as well as other similar previously conducted work (Gabora 1995; Corneli and Jordanous 2015; Pagnutti, Compton, and Whitehead 2016), study mostly the collaboration of agents that have an equal social status, in contrast to our case where the social status is hierarchical. Therefore we find that there's need for conducting the study presented in this paper to shed some light

into asymmetrical social relations in computational creativity.

Social Development

The master-apprentice approach gives us an intriguing test bed for modelling different social interactions between the master and the apprentice. With such a complex phenomenon as human social behavior, we are bound to limit our focus on a subarea of the phenomenon. In this section, we describe different psychological approaches in understanding socialization.

Socialization, i.e. becoming a part of a social group, is an important part of the psychological development of an individual. Even to such a degree that a child who is never exposed to other people will not develop a language nor an understanding of self. Socialization, thus seems to play a crucial role in higher-level cognitive development of everything that we consider to separate a man from an animal. Perhaps this great level of importance has been the reason a great many researchers have dedicated effort in unraveling this mystery.

The ecological systems theory of social development (Bronfenbrenner 1979) highlights the importance of bidirectionality of different social groups. An individual child is in the middle of the model, but just as the immediate close family affects on the child, the child is also an actor in the process of socialization. The theory identifies multiple different systems from close family all the way to the level of the society that play a role in the social development of a child. This theory is quite complex to model computationally.

A take, simpler to model, on the social development is that of parenting styles (Baumrind 1991). We find these findings more suitable as a starting point for modelling the socialization of the apprentice in our master-apprentice approach. The parenting styles can be divided into four main categories: authoritative, authoritarian, permissive and rejecting-neglecting. These categories deviate from each other on the two-fold axis of demandingness and responsiveness as seen in Figure 1.

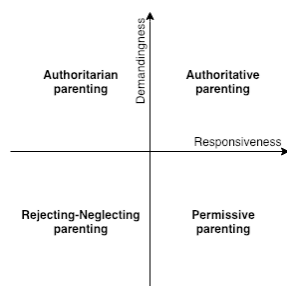


Figure 1: Parenting styles

The authoritative parents are high on both demandingness and responsiveness. They set rules, but the rules are negotiable. The parenting is more supportive than punitive in nature. The authoritarian parents, on the other hand, are low on

responsiveness and high on demandingness. They set non-negotiable rules and expect obedience without explanation.

The permissive parents are low on demandingness and high on responsiveness. They are very lenient and avoid confrontation. The rejecting-neglecting parents, however, are low on both axis. They hardly engage in parenting, they offer little support and do not set any rules.

Creativity

The original research on the master-apprentice approach (Alnajjar and Hämäläinen 2018) used the creative tripod (Colton 2008) to define creativity in general and in the context of their work on creating movie titles satirical towards Saudi-Arabia by following the notions of the SPECS approach (Jordanous 2012). We use the same creative tripod framework to adapt their definition into our similar task of creating movie titles with food puns. This definition provides us with a reasoned way of conducting evaluation of the overall creativity of our systems.

The creative tripod requires three key notions to be present in a system in order for it to achieve creativity. These are *skill*, *imagination* and *appreciation*. All of these components must be present simultaneously in a creative system, or the system will lack creativity.

For our systems to exhibit skill, they will need to take a movie title as input and produce a new one with a food pun. As in the case of the earlier master-apprentice approach, the new humorous title should still communicate the original title, i.e. the original name of the movie should be recognizable.

Requirements for a pun are that it reassembles the original word in pronunciation and that it is humorous. According to (Oring 2003), *incongruity* results in humour if it is delivered in a playful fashion and accompanied by its resolution. Another, maybe a bit more concrete way of looking at humour, is seeing incongruity as a *surprise* and resolution as *coherence* c.f. (Brownell et al. 1983).

Surprise in the context of humor means that the brain forms an expectation and this expectation is then broken by the humorous element of the pun. Such is the case in the pun *Harry Potter and the Deathly Marshmallows* where the surprise is caused by the fact that the expected word *Hallows* is replaced by *Marshmallows*. For the pun to be coherent, it should make sense in the context of the original movie. In this case, a thought of deathly marshmallows attacking the Hogwarts, although bizarre, can still be seen as coherent.

For achieving appreciation, the systems will need to be able to assess the humorousness of the created pun in terms of surprise, coherence and sound similarity. In addition to the humour, the system should be able to evaluate the recognizability of the original title.

We define imagination by using the dichotomy of creativity introduced by (Boden 2004). This way of understanding creativity divides it in two different types: P-creativity and H-creativity. P-creativity is the minimal requirement we set for imagination of the systems, and it means that a creative entity should be able to come up with something that is novel to itself. H-creativity, on the other hand, refers to an innovation that is novel in a more global scale, i.e. nobody else

has come up with a similar creative artifact before. While P-creativity is the minimum requirement, we consider H-creativity as a more desired requirement for imagination.

The Data

As our approach is to generate food related puns, we need a vocabulary consisting of food related terms. For this purpose we use the Historical Thesaurus of the Oxford English Dictionary¹. We use all the nouns recorded under the topic *food and drink* in the *external world* taxonomy. This list contains 15,314 different nouns.

We extract real movie titles from the IMDB² (Internet movie database). As we want our movie title corpus to consist only of well known movies, we want to filter out all the less known indie movies. To achieve this, we filter out movies that have received less than 100,000 votes, leaving us with 1,661 movie titles. For the master and apprentices this is further limited to 1276 titles by filtering out the titles that consisted only of one word.

For parallel humorous movie title data (later peer data), we crawl comments on an Instagram post for an entertainment account³. People were encouraged to come up with creative movie titles containing a pun related to food. The total number of comments crawled is 16,088⁴. Then, we follow the same approach applied in (Alnajjar and Hämäläinen 2018) to map the crawled data to movie titles. In summary, we preprocess the text to remove any hashtags and mentions, and then we measure the character and word edit distances between the comments and movie titles. Finally, a comment is considered to be a punny variation of the matched movie title with the least edit distance, only if it had at most three word differences while ensuring that there exist at least one word matching the movie title. This process yields 9,294 human-authored movie titles containing a pun.

The Master-Apprentice Model

The master-apprentice model consists of a computationally creative genetic algorithm that implements the criteria set for appreciation in its fitness function and an apprentice that is an NMT (neural machine translation) model. The master generates parallel data for the apprentice to learn from, while the apprentice can also learn from its peers. In our setting, we have four different apprentices; one for each parenting style.

Master

Inspired by the work on slogan generation presented by Alnajjar, Hadaytullah, and Toivonen (2018), we employ a similar generator to act as a master in our model. In our case, the generator, which is a genetic algorithm, receives an original movie title as input and outputs an entire population of movie titles carrying a pun, based on the input movie title. The master makes use of the food related vocabulary described earlier to replace words in the original title while

optimizing multiple parameters to increase the aptness of the substitution and the punniness of the title. The following subsections elucidate the algorithm.

Evolutionary algorithm The first step in the evolutionary algorithm is producing the initial population, which will go through the process of evolution during a certain number of generations. The evolutionary algorithm employed is a standard $(\mu + \lambda)$ ⁵ where mutation and crossover are applied to the current population to produce λ offspring. Individuals in the current population and their offspring are then evaluated by the algorithm to find the fittest μ number of individuals to survive to the next generation. Once the specified number of generations (10, in our case) is reached, the evolutionary process ends and returns the final population.

Initial Population The initial population consists of μ copies of the input movie title. For each copy, a randomly selected noun, adjective or verb is replaced with a random word from the vocabulary. We used *Spacy* (Honnibal and Montani 2017) to parse titles. We inflect the substituting words using *Pattern* (De Smedt and Daelemans 2012) to match the morphology of the original word when needed. The altered titles assemble the initial population.

Mutation and Crossover In our evolutionary algorithm, we implement one kind of mutation and crossover. The mutation process substitutes words in the individual in the same fashion as done in the creation of the initial population. The crossover employed is a standard single-point crossover, i.e. a random point in individuals is selected and words to the right of the point are switched between them.

Evaluation In our evaluation metric, we propose four internal evaluation dimensions to measure the fitness of an individual. These dimensions are (1) prosody, (2) semantic similarity to “food”, (3) semantic similarity to the original word, and (4) number of altered words. The first two dimensions are maximized, whereas the last two are minimized.

The prosody dimension is a weighted sum of four prosody sub-features, which are consonance, assonance, rhyme and alliteration. This dimension measures the sound similarity between the original word and its substitution. To measure the sound similarity, we use *espeak-ng tool*⁶ to generate IPA (international phonetic alphabet) transcriptions for assessing the prosody.

To measure the semantic similarity between two words, we employ a pre-trained *Glove* model⁷ with 6 billion tokens and a dimension size of 300. The model is trained on Wikipedia and English Gigaword Fifth Edition corpus. Using the semantic model, the next dimension computes the maximum semantic similarity of words in the title to the word “food”.

The third dimension measures the mean of the semantic similarity of new words to their original corresponding word. We minimize this dimension to increase surprise, with

¹<http://www.oed.com/thesaurus>

²Dumps from <https://datasets.imdbws.com/>

³<https://www.instagram.com/p/BsWki-PFbMO/>

⁴Crawled on the second of February

⁵We set both to 100 empirically.

⁶<https://github.com/espeak-ng/espeak-ng>

⁷<https://nlp.stanford.edu/projects/glove/>

the idea that a lower semantic similarity between the original word and its substitute would result in a bigger surprise.

The last dimension keeps track of the number of words modified in comparison to the original title. Minimizing this dimension motivates that less substitutions are made to the title, which makes it more recognizable.

These are the criteria based on which the fitness of individuals is evaluated at the end of each generation to let only the best ones survive to the next generation.

Selection and Filtering To reduce having a dominating dimension and motivate generating titles with diverse and balanced scores on all four dimensions, we opt for a non-dominant sorting algorithm *-NSGA-II-* (Deb et al. 2002) as the selection algorithm.

During each iteration of the evolution, the current population and its offspring go through a filtering phase which filters out any duplicate titles.

Final Verdict On top of individual evaluation metrics, we introduce master's final verdict, which is a way of telling whether the master likes the generated title. The final verdict of the master is a binary decision, i.e. an individual is either good or not. In practice, the final verdict is defined as conditional thresholds on each dimension. These thresholds are 1) a positive non-zero value for prosody, 2) a positive non-zero semantic similarity to "food", 3) a semantic similarity less than 0.5 of the new word to its original and 4) not more than 50 percent change of content words.

The master uses this functionality to express its liking to titles outside of its own creations such as those created by the apprentice. Whenever we talk about the master liking something in this paper, we mean that the final verdict has a Boolean value of true.

Apprentice

For the apprentices we use OpenNMT (Klein et al. 2017), which implements an RNN based sequence to sequence model. The model has two RNN encoding layers and two RNN decoding layers.

The attention mechanism is the general global attention formulated by (Luong, Pham, and Manning 2015). The difference to the OpenNMT default parameters in our system is that we use the copy attention mechanism which makes it possible for the model to copy words from the source. This is useful since the task is to translate within the same language.

All of the apprentice models described in this paper have been trained by using the same random seed to make their intercomparison possible.

Different Parenting Styles

We model computationally the four different parenting styles, authoritarian, authoritative, permissive and rejecting-neglecting, in the way the master interacts with the apprentice during the training process of the NMT model.

The training process is done iteratively. In each iteration, the apprentice is trained for 1000 training steps. After each iteration, the apprentice produces an output based

on the 1276 popular IMDB movie titles. This output is then evaluated by the master accordingly to the parenting style in question and adjustments are made to the training data based on the master's parenting. The apprentices are trained for 20 iterations.

The Authoritarian Master only lets the apprentice learn from its own output. The apprentice is not exposed to any of the peer data and the apprentice's own creations are not taken into account.

The Authoritative Master lets the apprentice learn from its own creations and those peers who it considers good enough by the final verdict (this means 2446 titles). The apprentice can show its creations to the master after each training iteration, out of which the master picks the ones it likes and adds them to the training material of the apprentice. The training of the NMT model continues with the modified corpus.

The Permissive Master lets the apprentice learn from its own creations and all of the peer data. When the apprentice presents its own creations at the end of a training iteration, the master praises them all and adds them to the training data.

The Rejecting-Neglecting Master does not care about the apprentice. The apprentice has no choice but to learn from its peers. The apprentice does not learn from its own creations because it receives no support from the master.

Training the Apprentice

The master is run once to create its own movie titles with food related puns. This parallel data of 8306 titles is shared across the different parenting styles. During the training process of the apprentice, the master does not generate new titles of its own, but only interferes in the selection of the parallel data used in the next training iteration as described in the sections above.

After each iteration, we calculate BLEU score (Papineni et al. 2002) and a uni-gram PINC score (Chen and Dolan 2011) for the outputs of the apprentices. We compare the outputs both to the training material coming from the master and the material from the peers. For each title generated by the apprentice, we take the maximum BLEU and minimum PINC score and take an average of them for each iteration.

BLEU score is traditionally used in machine translation to evaluate how good the final translation is in terms of a gold standard. We, however, do not use BLEU as a final evaluation metric, but rather use it to shed some light into how closely the outputs of the apprentices resemble those of the master or the peer written titles. BLEU measures the similarity, whereas PINC measures divergence from the original data. In other words, the higher the BLEU, the more closely the apprentice imitates and the higher the PINC the less it imitates the master or the peers.

As indicated by Figure 2, the authoritarian scenario, where the training data consists only of the master's output, starts quickly producing the output most similar to the master. Where as the authoritative scenario leads to a bit less similarity to the master. The effect of the peer data is

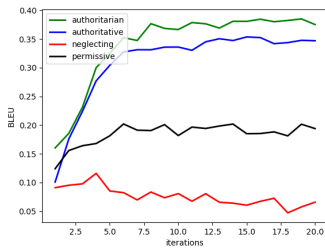


Figure 2: BLEU when comparing to the master

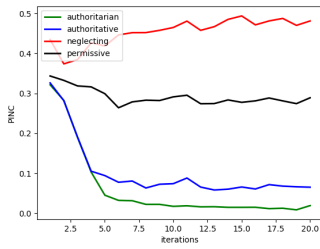


Figure 3: PINC when comparing to the master

very well visible in the permissive and neglecting scenarios. The PINC scores in Figure 3 show the other side of the coin where the authoritative and authoritarian scenarios are the least divergent and the permissive and neglecting ones the most divergent.

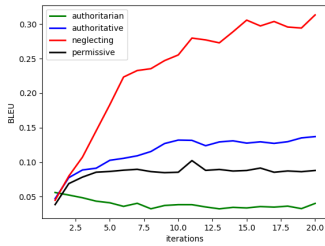


Figure 4: BLEU when comparing to peers

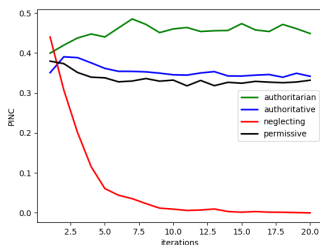


Figure 5: PINC when comparing to peers

When we do the BLEU comparison to the peer data as seen in Figure 4, we can see that only the neglecting scenario leads to high similarity with the peers, where as the other scenarios are still quite low, the lowest being the authoritarian scenario. The PINC scores tell a similar story in Figure 5, where the neglecting scenario leads to the least amount of divergence, leaving the authoritarian scenario the most divergent.

Results and Evaluation

In this section, we show some of the results produced by the different systems. In addition, we evaluate the different parenting style scenarios after each iteration with the master's appreciation function. Later, an evaluation is conducted by humans.

Results and Master's Liking

Results from the approaches can be seen in Table 1. The master did not produce any training data for the last two titles in the examples. Looking at these results qualitatively, in broad lines, the permissive and neglecting scenarios produced worse output than the ones exposed to the master's training data. The apprentice exposed to authoritarian parenting struggles in producing output for titles not present in the training data. The authoritative scenario leads to the most consistent results. The quantitative human evaluation in the next section is used to verify these initial observations.

Another way to look at the results is to use the appreciation metrics implemented in the master. Figure 6 shows the percentage of how many titles the master liked after each training iteration.

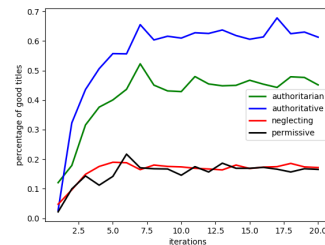


Figure 6: Master's liking of the output

As we can see, the appreciation the master has ranks the authoritarian and authoritative scenarios higher than the permissive and neglecting ones. Even in the authoritarian case, the master does not like all of the output produced by the apprentice, which shows that the appreciation learned by the apprentices is different from the one implemented in the master.

It is interesting to see to what extent the master's liking correlates with the evaluation results of the human judges. This can reveal more information about the adequacy of the appreciation of the master in this creative task. Or does the master's appreciation only tell about obedience when applied to the apprentices' output?

original	master	authoritarian	authoritative	permissive	neglecting
the butterfly effect	<i>the brewery effect</i>	<i>the butterfly kimchi</i>	<i>the butterfly chicken</i>	<i>the butterfly effect</i>	<i>the lasagna effect lazarus</i>
how to train your dragon	<i>how to train your pepperoni</i>	<i>how to train your avocado</i>	<i>how to train your pepperoni</i>	<i>how to train your bacon</i>	<i>how to train your bacon</i>
fantastic beasts and where to find them	—	<i>fantastic ordinary and where to find</i>	<i>fantastic beets and where to find them</i>	<i>fantastic beefs and where to find them</i>	<i>fantastic beets and where to find them</i>
under the skin	—	<i>under the cereals</i>	<i>under the silver cake</i>	<i>under the 13th</i>	<i>fryday the 13</i>

Table 1: Examples of the final output of the different models

Evaluation Questions

In this section we provide some reasoning in our selection of the evaluation questions that are presented to the human judges. Earlier, we defined the creativity in the case of pun generation using the creative tripod as our theoretical framework. This means that on a higher level, our evaluation questions should evaluate *skill*, *appreciation* and *imagination*.

Skill Our definition for skill stated that the system should be able to take an existing movie title and produce a food related pun as an output. A further requirement was that the original title should be recognizable from the generated one.

1. The title has a pun in it
2. The title is related to food
3. The original title is recognizable

The evaluation questions described above are designed to evaluate the requirements set for skill. We evaluate whether a pun is perceived and whether the new title relates to food separately, as it might be that the replacement word delivers a pun, but is not food related or vice-versa.

Appreciation We defined appreciation from the humor stand point. A good title with a pun is also funny. For something to be funny, i.e. humorous, the pun has to exhibit coherence and surprise.

4. The title is humorous
5. The pun is surprising
6. The pun makes sense in the context of the original movie

We choose to evaluate the overall humor value of the title separately from the components that constitute it. The last two questions are designed to evaluate surprise and coherence respectively.

Imagination We used Boden's dichotomy to establish the definition of imagination. The minimal requirement was set to P-creativity. However, P-creativity can easily be verified by looking at the training data and the final output, if the output is different from the training material, there is P-creativity. Therefore, we use human judges to assess the H-creativity of the outputs.

7. The pun in the title is obvious
8. The pun in the title sounds familiar

If the pun is obvious, it probably is not too H-creative, as an obvious pun could be said by just about anyone, also if the pun sounds familiar, it has probably been said by someone before.

Human Evaluation

We take a random sample of 20 original movie titles that were only present in the training data provided by the master, 20 titles that were only present in the peer data and 20 titles that were in both sources of parallel data. We evaluate the creative output of each apprentice for these randomly sampled titles. In addition, we evaluate the master's output for the 40 titles of the sample it had generated movie title puns for. As the master has generated multiple creative titles per original title, we pick one randomly for each original title. Altogether, we are evaluating 280 computer created titles.

The evaluation was conducted on a crowd-sourcing platform called Figure Eight⁸. The platform assigned people to conduct evaluation in such a way that each title was evaluated by 35 different users. The users could choose how many titles they wanted to evaluate. The results of the evaluation are show in Table 2. In the Training column, *both*, *peer only* and *master only* indicate whether the original title was only present in the master produced training data, peer produced training data or in both respectively.

The authoritarian scenario didn't get the best average score for any of the test questions and neither did the master. They both score particularly low on the Q2, which reflects the fact that some of the words in the HTOED food and drink taxonomy were only loosely related to food such as *steam* and *spit*. It is interesting to note that the authoritarian scenario gets the best results for Q3, Q6 and Q7 for titles it did not encounter in the training data, in other words it has developed an appreciation of its own that does not just mimic what the master produces and fail otherwise. In light of these results, we can deduce that the master produced worse titles with food puns than real people, which left both the master and the authoritarian scenario without the first place on any of the test questions.

The authoritative scenario, which was the highest ranking one according to master's liking as seen in Figure 6, got the best results for Q1 and Q5. This means that it succeeds the best in the main task of generating puns and they end up being the most surprising ones. It is also the only one that produces consistently good results (above 3 on the average) for all training test sets for Q1-Q6, unfortunately the results for Q7 and Q8 are also above 3 on the average meaning that it does not rank high on H-creativity.

The same consistency can not be perceived in the the per-

⁸<https://www.figure-eight.com/>

Style	Training	Q1		Q2		Q3		Q4		Q5		Q6		Q7		Q8	
		μ_x	SD	μ_x	SD	μ_x	SD	μ_x	SD	μ_x	SD	μ_x	SD	μ_x	SD	μ_x	SD
authoritarian	both	3.35	1.08	2.65	1.26	3.33	1.06	3.02	1.10	3.08	0.99	3.03	1.01	3.16	0.99	3.10	1.03
	peer only	2.97	1.18	2.14	1.17	3.44	1.13	2.66	1.15	2.82	1.08	3.10	1.11	3.03	1.12	3.11	1.13
	master only	3.34	1.07	2.89	1.28	3.30	1.05	3.00	1.12	3.07	1.05	3.04	1.02	3.12	1.02	3.07	1.04
authoritative	both	3.43	1.08	3.09	1.31	3.28	1.12	3.08	1.16	3.08	1.05	3.02	1.05	3.22	1.06	3.13	1.05
	peer only	3.41	1.14	3.23	1.35	3.37	1.21	3.13	1.16	3.07	1.08	3.08	1.09	3.24	1.10	3.16	1.13
	master only	3.47	1.03	3.15	1.33	3.41	1.08	3.17	1.11	3.16	1.02	3.16	1.04	3.28	1.01	3.24	1.06
master	both	3.38	1.07	2.79	1.28	3.29	1.06	3.07	1.12	3.09	1.04	3.10	1.06	3.22	1.02	3.14	1.05
	master only	3.40	1.07	2.61	1.30	3.34	1.11	3.10	1.15	3.09	1.02	3.04	1.03	3.17	1.04	3.11	1.06
neglecting	both	3.45	1.11	3.28	1.32	3.34	1.11	3.28	1.12	3.16	1.06	3.12	1.04	3.21	1.08	3.22	1.07
	peer only	3.36	1.07	3.02	1.37	3.31	1.11	3.12	1.15	3.09	1.02	3.14	1.04	3.19	1.02	3.14	1.06
	master only	3.28	1.13	2.87	1.35	3.34	1.12	3.09	1.14	3.05	1.05	3.07	1.06	3.15	1.06	3.18	1.08
permissive	both	3.23	1.18	2.67	1.38	3.59	1.06	3.06	1.13	3.08	1.08	3.30	1.04	3.21	1.07	3.30	1.10
	peer only	3.05	1.19	2.87	1.39	3.25	1.18	2.88	1.13	2.88	1.09	3.00	1.08	2.99	1.11	3.04	1.14
	master only	3.09	1.23	2.32	1.24	3.64	1.11	2.88	1.15	2.91	1.12	3.07	1.15	2.98	1.13	3.04	1.12

Table 2: Mean and standard deviation.

Style	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8
authoritarian	78.33%	31.67%	83.33%	31.67%	48.33%	68.33%	70.00%	68.33%
authoritative	93.33%	60.00%	83.33%	56.67%	63.33%	66.67%	90.00%	70.00%
master	92.50%	37.50%	87.50%	62.50%	60.00%	65.00%	80.00%	67.50%
neglecting	86.67%	60.00%	81.67%	66.67%	65.00%	73.33%	75.00%	78.33%
permissive	66.67%	33.33%	85.00%	43.33%	46.67%	73.33%	68.33%	71.67%

Table 3: Percentage of movie titles having an average score by judges greater than 3

missive case as scores below 3 are common across the test questions. It however, manages to score the best for Q3 and Q7-Q8, in other words, it can achieve the best H-creativity and the original titles can be the most recognizable, although not consistently so. This shows, that even though the appreciation the master has might not be spot on, as it is not able to produce the best scoring titles, having moderation on the peer data and critical assessment of the apprentice generated results during the training by the master, has a positive effect on the consistency of the results. In the permissive scenario, the apprentice was exposed to everything without criticism and in the authoritative some criticism was used to filter the training data, which made the authoritative scenario more consistent, but less H-creative.

Finally, the neglecting scenario gets the best scores for the Q2, Q4 and Q5. It is the best one at producing humorous, surprising and food related titles. It is quite consistent with only the results for Q2 with previously unseen titles giving a score that is inferior to 3. The good results of the neglecting scenario serve as an additional proof to the fact that the output of the master is worse than human written titles.

Table 3 shows the results from another stand point. The table shows overall how many titles got the average rating above 3 for each test question. These numbers are in line with what was previously discussed about the Table 2. The authoritarian scenario leads to the worst performance, but this time master gets the highest percentage point of titles above 3 for Q3. In the authoritative scenario most of the titles have a clear pun and are related to food with the highest percentage point. The permissive scenario holds the best percentage points for Q6 and Q7. And the neglecting gets the best percentage points in Q2, Q4, Q5 and Q6.

Discussion and Future Work

The evaluation results were not completely in line with what we can observe by looking at the titles output by the different methods by ourselves. This raises the question whether our definition for creativity in movie title puns is adequate and whether the evaluation questions we formulated based on the definition really measure what they were designed to measure. Because we have worked with a clear definition for creativity in this paper, it is possible to take this under a critical study in the future. We also find evident that qualitative research on the output titles with respect to the quantitative results we got from the human judges is needed to evaluate the evaluation itself.

Having a master with appreciation filter the parallel data of the apprentice was beneficial for consistency (see authoritative vs permissive). Although the evaluation results showed that the appreciation is not in par with that of a real human, the implication remains that a good external appreciation can be beneficial for the learning outcome of the apprentice model. As we used a rather generic NMT model for the apprentice, our findings might be of a use in more traditional context of sequence-to-sequence models such as machine translation, text summarization or paraphrasing.

For now, the master and apprentice have been studied in a social vacuum, where peer data is the only link to the surrounding world. However, in the future it would be fruitful to see how the creative outcome changes when the master and the apprentice are exposed to a more complex social system such as the one described by Bronfenbrenner (Bronfenbrenner 1979). In such a society, the master would also be under a social pressure in changing its own standards of appreciation.

Conclusions

This work has presented one of the first contributions to the field of computational social creativity where the computationally creative agents are in a hierarchical social relation. This asymmetry offers an intriguing setting for studying socialization of computational agents from the creativity perspective.

Despite building our definition of creativity upon an existing theory and formulating the test questions based on the definition, the quantitative evaluation left many questions unanswered. The results presented in this paper call for qualitative evaluation to understand the phenomenon of evaluation in this particular context.

Nevertheless, our findings suggest that having appreciation in parenting, or training, an NMT model can be of a benefit. The applicability of these findings into sequence-to-sequence deep learning models in a more generalized fashion is an interesting research question on its own right.

References

- Alnajjar, K., and Hämäläinen, M. 2018. A Master-Apprentice Approach to Automatic Creation of Culturally Satirical Movie Titles. In *Proceedings of the 11th International Conference on Natural Language Generation (INLG)*, 274–283.
- Alnajjar, K.; Hadaytullah, H.; and Toivonen, H. 2018. “Talent, Skill and Support.” A method for automatic creation of slogans. In *Proceedings of the Ninth International Conference on Computational Creativity*, 88–95.
- Baumrind, D. 1991. Parenting styles and adolescent development. *The Encyclopedia of Adolescence* 758–772.
- Boden, M. A. 2004. *The creative mind: Myths and mechanisms*. Routledge.
- Bronfenbrenner, U. 1979. *The ecology of human development*. Harvard university press.
- Brownell, H. H.; Michel, D.; Powelson, J.; and Gardner, H. 1983. Surprise but not coherence: Sensitivity to verbal humor in right-hemisphere patients. *Brain and Language* 18(1):20 – 27.
- Chen, D. L., and Dolan, W. B. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, 190–200.
- Colton, S. 2008. Creativity Versus the Perception of Creativity in Computational Systems. In *AAAI Spring Symposium: Creative Intelligent Systems*, Technical Report SS-08-03, 14—20.
- Corneli, J., and Jordanous, A. 2015. Implementing feedback in creative systems: a workshop approach. In *Proceedings of the First International Conference on AI and Feedback-Volume 1407*, 10–17. CEUR-WS. org.
- De Smedt, T., and Daelemans, W. 2012. Pattern for Python. *Journal of Machine Learning Research* 13:2063–2067.
- Deb, K.; Pratap, A.; Agarwal, S.; and Meyarivan, T. 2002. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation* 6(2):182–197.
- Gabora, L. 1995. Meme and variations: A computer model of cultural evolution. *1993 Lectures in Complex Systems* 471–486.
- Hantula, O., and Linkola, S. 2018. Towards goal-aware collaboration in artistic agent societies. In *Proceedings of the Ninth International Conference on Computational Creativity*.
- He, H.; Peng, N.; and Liang, P. 2019. Pun generation with surprise. *arXiv preprint arXiv:1904.06828*.
- Honkela, T., and Winter, J. 2003. *Simulating language learning in community of agents using self-organizing maps*. Helsinki University of Technology.
- Honnibal, M., and Montani, I. 2017. spaCy 2: Natural Language Understanding with Bloom Embeddings, Convolutional Neural Networks and Incremental Parsing. *To appear*.
- Jordanous, A. 2012. A standardised procedure for evaluating creative systems: Computational creativity evaluation based on what it is to be creative. *Cognitive Computation* 4(3):246–279.
- Klein, G.; Kim, Y.; Deng, Y.; Senellart, J.; and Rush, A. M. 2017. OpenNMT: Open-Source Toolkit for Neural Machine Translation. In *Proc. ACL*.
- Linkola, S.; Takala, T.; and Toivonen, H. 2016. Novelty-seeking multi-agent systems. In *Proceedings of The Seventh International Conference on Computational Creativity*.
- Luong, M.-T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.
- Oring, E. 2003. Engaging humor. *Urbana and Chicago: University of Illinois Press*.
- Pagnutti, J.; Compton, K.; and Whitehead, J. 2016. Do you like this art i made you: introducing techne, a creative artbot commune. In *Proceedings of 1st International Joint Conference of DiGRA and FDG*.
- Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, 311–318.
- Ritchie, G. 2005. Computational mechanisms for pun generation. In *Proceedings of the Tenth European Workshop on Natural Language Generation (ENLG-05)*.
- Saunders, R., and Bown, O. 2015. Computational social creativity. *Artificial life* 21(3):366–378.
- Yu, Z.; Tan, J.; and Wan, X. 2018. A neural approach to pun generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 1650–1660.

Towards a General Framework for Humor Generation from Rated Examples

Thomas Winters and Vincent Nys and Danny De Schreye

Computer Science Department

KU Leuven

Leuven, Belgium

{firstname}.{lastname}@cs.kuleuven.be

Abstract

Many computer systems are becoming increasingly tailored to their users, customizing and optimizing their experience. However, most conversational agents do not follow this trend when it comes to humorous interactions. Instead, they employ pre-written answers regardless of whether the user liked previous similar interactions. While there already exist several computational humor systems that can successfully generate jokes, their joke generation models, parameters or even both are often fixed. In this paper, we propose GOOFER, a general framework for computational humor that learns joke structures and parameterizations from rated example jokes. This framework uses metrical schemas, a new notion we introduce, which are a generalization of several types of other schemas. This new type of schema makes regular schemas compatible with machine learning techniques. We also propose a strategy for identifying useful humor metrics based on humor theory, which can be used as features for the machine learning algorithm. The GOOFER framework uses these novel concepts to construct a pipeline with new components around previous generators. Using a mapping to our previous work on analogy jokes, we show that this framework cannot only generate this type of jokes well, but also find the importance of specific humor metrics for template values. This indicates that it is on the right track towards joke generation systems that can automatically learn new templates and schemas from rated examples. This work thus forms a stepping stone towards creating programs with a sense of humor that is adaptable to the user.

Introduction

Generating jokes is one of the research tasks in the field of computational humor. In this field, there are three distinct kinds of computational tasks, being the generation, detecting and analysis of humorous artefacts (Binsted et al. 2006; Ritchie 2002). The ideal computational humor system would be able to perform all three mentioned task categories on all types of humor. However, most computational humor systems tend to focus on a single task, performed on a specific type of humor (Ritchie 2001). Joke generation systems also tend to follow a fixed-rule set, and are thus unable to nudge their jokes towards the preferences of a certain user. Existing systems also exist in isolation from each other. Until now, no attempt has been made to generalize over the existing research in the field in order to create systems that

generalize previous research and as such are capable of doing more types of tasks on more types of humor. This paper focuses on the first and the last task categories: generating and analysis of humor. This allows the generator to steer towards certain jokes based on analysis of rated jokes (e.g. of a certain user). In previous work, we explored how to perform this task on analogy jokes using a system called GAG (*Generalized Analogy Generator*) (Winters, Nys, and De Schreye 2018). In this work, we extend that system to a general framework called GOOFER (*Generator Of One-liners From Examples with Ratings*). This framework could be used for many types of short jokes and is able to generalize and improve several previous humor generators.

The effectiveness of a joke generation system depends on the quality of the jokes it produces. However, this quality depends on many factors, e.g. word choice (Stock and Strapparava 2003), word order (narrative) (Raskin 1985), amount of incongruity (Kao, Levy, and Goodman 2016; Ritchie 2002) and cultural understanding (Petrović and Matthews 2013). Some promising factors have not yet been studied in much detail, such as the individual and temporal dependencies, by which we mean that joke quality also depends on the observer and the time when the joke was observed. It goes without saying that to increase the effectiveness of a joke generation system, the jokes should be tailored to an individual user's preferences. Seeing the disagreement between users in the evaluations of jokes from previous research (Goldberg et al. 2001; Petrović and Matthews 2013; Stock and Strapparava 2003; Winters, Nys, and De Schreye 2018), it is necessary to account for individual user preferences when generating jokes. However, most systems that are able to adapt jokes to user preference, are not responsible for the generation of these jokes, but are mere humor recommendation engines (Goldberg et al. 2001). We assume the reason for this absence is that most of the existing humor generators utilize some form of rule set or assumptions about their type of joke, or have a model trained to mimic the textual input without considering the quality. This implies that they cannot update the content of their jokes without human intervention. Even worse, this also implies that most are completely incapable of automatically learning new types of humor than the type of humor they were designed for, nor can they update their rules based on rating for previously generated jokes. This has some serious impli-

cations for both the individual and temporal dimensions of the generator’s joke quality.

Having the capability of learning humor from joke examples along with their perceived quality can help joke generators nudge their generative space towards more interesting generations. Using this mechanism, it can account for the temporal and individual factors of humor appreciation, by learning new types of jokes from popular platforms and learn to adapt to users by feeding its own generated jokes with ratings back into its training data.

Background

Humor Theory

The incongruity-resolution theory (IR) is probably the most widely accepted theory, and is also most relevant to computational humor research (Krikmann 2006). It states that humor originates from noticing an incongruity in incompatible views, and resolving this incompatibility (Binsted et al. 2006). This is in line with the argument that humor stems from a sudden mental bisociation. (Koezler 1964). A “bisociation” describes the two different viewpoints an act of creativity tends to have, and manifests itself as a jump between two self-consistent but incompatible frames of reference (Ritchie 2001; Krikmann 2006). Most humor theorists seem to agree on the idea of humor being the combination of two such frames, since most jokes can be distilled to a set-up and a punchline (Ritchie 2001).

Ritchie argued that several formal humor theories were often far from being implementable enough for generative purposes, and created his own formal theory, extending the incongruity-resolution theory. He uses the notion of surprise disambiguation, which states that two different interpretations for the set-up of a joke must exist, an obvious interpretation (e.g. “*the fish are in an aquarium*” for the joke in Figure 1) and a hidden interpretation (“*the fish are in a military vehicle*”) that only becomes apparent through the punchline, which forces the hidden interpretation as the only remaining possible interpretation (Ritchie 1999). Hearing the punchline will thus cause an inconsistency with the interpretation assumed when hearing the setup. This inconsistency starts a mental search process to the hidden interpretation, which should be the only interpretation compatible with both the set-up and the punchline. Finding this cognitive rule to explain the mismatch causes laughter to ensue (Ritchie 1999).

Ritchie proposed several properties to identify the relationships specified in his theory, which creates more concrete measurements for joke generation and detection than previous theories (Ritchie 1999; 2002), which we visualized in Figure 1.

- **OBVIOUSNESS:** The first interpretation should be very obvious (e.g. *fish are usually in aquariums*), otherwise observers hearing the hidden interpretation will not experience an incongruity at the end of the joke, due to the joke being fully compatible with this interpretation. This property thus quantifies how obvious the initial interpretation of the set-up is.
- **CONFLICT:** There should be a conflict between the punchline and the obvious interpretation, otherwise the

search process to the cognitive rule explaining the mismatch will not start (e.g. *you can not drive an aquarium*).

- **COMPATIBILITY:** The meaning of the punchline should be compatible with the second, hidden interpretation, otherwise the search for a cognitive rule will not stop, ensuing in puzzlement (e.g. “*drive tank*” is compatible with “*in a military vehicle*” interpretation).
- **COMPARISON:** There should be a contrasting relationship between the two possible interpretations of the setup, as there would be no bisociation otherwise (e.g. *if the hidden interpretation was “in the sea”, it would be less funny*).
- **INAPPROPRIATENESS:** The second interpretation should be inherently odd, inappropriate or taboo, as this will make the interpretation less obvious, and more humorous to the observer (e.g. *fish usually do not drive vehicles*).

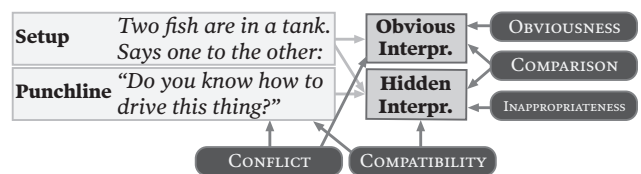


Figure 1: Visualisation of the IR theory

Templates & Schemas

There are several approaches to text generation, such as templates, grammars, Markov chains and recurrent neural networks. Templates are probably one of the most simplistic and naive methods, but they are a powerful tool mostly employed in macros, user interfaces and chat bots (Pilato et al. 2008). They are also extensively used in computational humor projects (Binsted and Ritchie 1994; Manurung et al. 2008; Venour 1999; Lessard and Levison 1992; Raskin and Attardo 1994; Winters and Mathewson 2019).

A template, in the meaning we intend, can be defined as a text with slots or variables. These variables are filled in later by another data source. It also allows data sources to work with different templates. In this work, we call the values to be filled into a particular template “template values”.

Schemas are often used as the data source for templates in computational humor (Binsted and Ritchie 1994; Manurung et al. 2008; Venour 1999). They are for example used in the first computational joke production engine, JAPE, as well as its successor STANDUP (Manurung et al. 2008). These systems generate punning riddles in a question-answer format. In these works, schemas are defined as the structure defining the relationships between key words in a joke (Binsted and Ritchie 1994).

In STANDUP, schemas consist of five parts (Manurung et al. 2008):

- **Header:** the variables and the name of the template, e.g. *newelon2 (NP, A, B, HomB)*.
- **Lexical preconditions:** the syntactic, phonetic, structural or semantic constraints on the

variables, e.g. `nouncompound (NP, A, B)`, `homophone (B, HomB)`, `noun (HomB)`.

- **Question specification:** the templates to match the assigned variables to, along with some lexical constraints for their template values, e.g. “*What do you call a [SynHomB] with a [MerA]?*” with constraints like `shareproperties (NP, HomB)`.
- **Answer specification:** Similar to the question specification, e.g. “*A [A] [HomB].*” with constraints like `phrase (A, HomB)`
- **Keywords:** used to define equivalence between jokes, e.g. `[NP, HomB]`.

Such a schema can then generate jokes like Joke 1.

JOKE 1:

*What do you call a shout with a window?
A computer scream. (Manurung et al. 2008)*

Unsupervised Analogy Generator

Petrović & Matthews have created a model for generating analogy jokes using the “*I like my X like I like my Y, Z*” template, which we will call the unsupervised analogy generator (Petrović and Matthews 2013). They argue their program is the first fully unsupervised humor generation system, as they did not rely on a hard-coded schema approach, but on relations used in a minimization model (although one could also argue that due to specifying this model, it is only partially unsupervised). Their model encodes five relations about the *X*, *Y* and *Z* in these analogy jokes. It fixes the template values such that every template value is a single word, more specifically that *X* and *Y* are both nouns and that *Z* is an adjective. The system requires *X* to be defined by the user, and uses n-grams to choose *Y* and *Z* in such a way that *Z* is an adjective usable for both *X* and *Y*. The relational assumptions used in the model are that the joke is better the more frequent the attribute is used to describe both nouns, the less common the attribute is, the more ambiguous the attribute is and the more dissimilar the two nouns are (Petrović and Matthews 2013). These assumptions are all shown to be implementable as a metric resulting in a number. In order to rank how funny a joke is, the program minimizes the product of these five metrics. This research thus does not use machine learning techniques on training data to generate jokes.

Evaluators considered jokes created by the unsupervised analogy joke generator funny 16% of the time. Human-produced jokes using the same template were considered to be funny in 33% of the time. (Petrović and Matthews 2013). A joke generated by this system can be seen in Joke 2.

JOKE 2:

*I like my relationships like I like my source, open
(Petrović and Matthews 2013)*

Joke Template Extraction

T-PEG (*Template-Based Pun Extractor and Generator*) is a system created for the extraction of templates, aimed at pun

templates (Hong and Ong 2009). It generates punning riddles similar to jokes created by JAPE and STANDUP. In order to find a template, the system receives a single punning riddle, for which it replaces some words with variables. The template extraction algorithm is capable of detecting several types of variables, even hidden schema variables. However, they noted that the system heavily relied on linguistic relationships between these template values. In the author’s evaluation, 69.2% of the found templates were actually usable for joke generation (Hong and Ong 2009).

Other researchers tested T-PEG by clustering several similar STANDUP-generated jokes based on structural similarity (Agustini and Manurung 2012). Their system extracts templates using T-PEG, and employs agglomerative clustering on these templates using a single majority rule using a semantic similarity evaluation function. They tested this system by automatically verifying whether the templates and schemas used in STANDUP generated jokes were correctly found, and found that it had an overall precision of 61% (Agustini and Manurung 2012).

GOOFER

GOOFER (“*Generator of One-Liners From Examples with Ratings*”) is a novel theoretical computational humor framework for generating short jokes based on rated example jokes. In this section, we generalize the notion of schemas, create a theoretically founded set of metrics for humor purposes, and describe the flow and components of this novel framework.

Schema Generalization

Constraint-based Schemas As mentioned earlier, templates and schemas are an often used approach in computational humor. Schemas usually use lexical relations on the single word template values, such as synonymy and homonymy. We call this type *constraint-based schemas*. Generating template values given a seed is straightforward in constraint-based schemas: once a template variable is filled in, the possibilities for the other words are limited to those that are in the strictly defined relations with the already filled in template values. The limited search space of constraint-based approaches has two effects: it has the benefit of being efficient when generating, but their generative space might appear small compared to generators using other approaches. As illustrated earlier, this type of schemas can be written in a ProLog-like notation, which reveals that these schemas can both generate as well as check jokes. This notation only works for constraint-based schemas, and a different notation for schemas is required in order to incorporate metrics instead of strict relations.

Schemas Generalization Recognizing components of joke generation systems as templates and/or schemas, even if they do not use these explicitly, is a useful exercise to come up with new ways of modeling similar systems. As such, Venour (1999) showed how a Tom Swifty joke generator (Lessard and Levison 1992) was implicitly using templates and schemas. We show how to extend his approach even further, and map other systems that do not use

constraint-based approaches onto a more general type of schema that would allow the use of machine learning algorithms. In order to achieve this, we introduce the notion of a metrical schema.

Metrical Schema We define a metrical schema as having the following components, inspired by the definition of a schema of the STANDUP generator (Manurung et al. 2008):

- **Header:** the name of the schema, as well as the variables used in this schema.
- **Generator:** a generator to propose candidate template values for jokes.
- **Features:** the features used and which variables they are used on. They map template values to numbers.
- **Aggregator:** the way to aggregate the features and to choose the output jokes, e.g. all feature values must be above a certain value or the sum of certain feature values is higher than the sum of other features.
- **Template:** the template with slots for the variables.
- **Keywords:** the most relevant variables, used for calculating equivalence between schema outcomes, and to avoid producing similar jokes which might bore the user due to lack of surprise.

Proof of Generalization In order to show that a metrical schema is a generalization of the constraint-based schema, we have to show that a theoretical mapping from the latter to the former exists. First, the header and the keywords are transferable. Second, multiple templates (such as in STANDUP) can be mapped to a single template through concatenation. Third, the constraints can be mapped to functions that output 0 if the given constraint is violated and 1 if it is satisfied. These functions form the features of the new metrical schema. Fourth, since the schema only allows the assignment of variables that make all the functions map to 1, the aggregator function is defined as a function that only returns *true* if all features return 1. The only attribute left to define now is the generator, as this came for free using the ProLog-like constraints. Since this is a theoretical mapping, we ignore efficiency. We can then define the generator as generating all possible combinations of assignments to the variables using all words of the constraint-based generator’s lexicon. This concludes the mapping from a constraint-based schema to a metrical schema, showing that it is a strictly more general notation.

Example of Mapping With this new notion of a metrical schema, we can map the previously discussed analogy generator (Petrović and Matthews 2013) to the template and schema approach. As discussed earlier, this system uses metrics to calculate five metric values from a joke using the “*I like my X like I like my Y, Z*” template. It applies minimization of the product of several values (e.g. dissimilarity and ambiguity of certain template values) over the space of possible jokes. This could thus not have been represented using constraint-based schema, as there is no notion of minimization nor feature values in this type of schemas. Their

system only generates “*I like my X like I like my Y, Z*” jokes, and has one model to generate this, implying it only uses one template with only one schema. It is relatively straightforward to map their model to a metrical schema, as can be seen on Figure 2. This mapping shows that the new metrical schema notion is also generalizing joke generators that did not explicitly use (constraint-based) schemas.

```

Header: pm_analogy_model(X, Y, Z)
Metrics:      relatedness(X, Z),
               relatedness(Y, Z),
               dissimilarity(X, Z), ambiguity(Z),
               uncommonness(Z)
Aggregator: Product of features is below threshold t.
Generator:
1. Take X from input.
2. Generate Z from X as a possible adjective used with
   X with Google Ngrams.
3. Generate Y from Z as a possible noun used after Z
   with Google Ngrams.
Template: I like my <X> like I like my
<Y>, <Z>.
Keywords: [X, Y, Z]

```

Figure 2: Our mapping of the unsupervised analogy generator (Petrović and Matthews 2013) to a metrical schema.

Classification and Regression Schemas This new generalization allows for the use of machine learning by choosing a machine learning algorithm for the aggregator component. A classification algorithm can learn which feature values are correlated with what discrete rating (e.g. from the mode score of a collection of ratings, or the score of one specific person). In a similar fashion, a regression algorithm can estimate a non-discrete rating (e.g. from the average rating). By training to distinguish good jokes from bad jokes, and only allow jokes with an estimated score above a certain threshold, these algorithms act as the aggregator of the metrical schema. They also need to be accompanied by a more naive generator as the template values generator of the metrical scheme. The classification algorithms thus judge whether any of the candidates generated by the template values generator have a score exceeding a certain threshold in order to be considered “good”. We call this type of metrical schema a *classification schema* or a *regression schema* depending on the used algorithm.

Metric Set Identification

Now that we have defined schemas such that they are capable of using classification and regression algorithms, we need to define the metrics usable for calculating features from template values. These metrics should be metrics that make sense for a joke judging algorithm. As discussed earlier, Ritchie’s incongruity-resolution theory identifies five properties necessary for verbal humor (Ritchie 1999). We can use these properties to identify and validate a set of potential metrics to identify humor, similar to what we did in GAG (Winters, Nys, and De Schreye 2018).

For OBVIOUSNESS, metrics need to measure how obvious an interpretation is. This can be approximated using association or semantic distances in a lexicon: words that are not far from each other semantically, are probably linked to the same and common (since a lexicon contains it) interpretation. Another good measuring function is word frequency using 1-grams. If the word frequency is high, chances are that this word is a common word, where people associate this word with one specific, obvious meaning more easily.

For CONFLICT, we need the first interpretation to conflict with the punchline. This can also be approximated using association or semantic distance, as larger distances correlate with higher conflict. Previous research used n-grams for this (Petrović and Matthews 2013), because when particular words of the punchline are used more with specific other words of the setup, the meaning of this combination of words suddenly becomes more important than all other used words linked to the first interpretation, increasing the conflict with these words.

For COMPATIBILITY, the metrics should measure how compatible the hidden interpretation is with the set-up, which causes the search process to stop searching for another cognitive rule. Again, n-grams can approximate this, as more frequent particular words indicate higher compatibility. The number of meanings a word has is another interesting metric related to its ambiguity (Petrović and Matthews 2013). The surprise disambiguation confirms this, as the set-up is supposed to be ambiguous, with one obvious meaning. Previous research also used metrics determining how similar words sound (Manurung et al. 2008; Binsted and Ritchie 1994; Valitutti et al. 2013; Venour 1999), as homonyms can link the first and the second interpretation, but ensure that only the second interpretation is appropriate in a context.

For COMPARISON, the metrics should measure how much contrast the two possible interpretations have. Previous research did this by analyzing the domains of the words (Raskin 1985) using *WordNet Domains* (Magnini and Cavaglia 2000; Stock and Strapparava 2003), although *WordNet* itself could also be used to find dissimilarity using semantic distance. Adjective vector differences have also been successfully used in computational humor research for approximating this property (Kiddon and Brun 2011; Petrović and Matthews 2013). Calculating this value is done by looking up the frequency of the adjectives used for a noun, and calculating a value based on its difference that describes how different the contexts are that certain words occur in.

For INAPPROPRIATENESS, the metrics should approximate how odd, inappropriate, taboo and/or absurd the second interpretation is. Adjective sexiness and noun sexiness are used in DEViANT innuendo detection system to calculate how likely it is that a word is an innuendo-related word (Kiddon and Brun 2011). It compares the frequency of the word in a sexual corpus with a non-sexual corpus for adjectives, and adjective vector differences with body parts for nouns. Inappropriateness can also be approximated using the unigram frequency in a balanced corpus, as it is related to its unpredictability, and is thus capable of identifying odd

words (Petrović and Matthews 2013).

The identified metrics form a foundation for the knowledge base of our generic framework and are used by the machine learning algorithms to extract features from given jokes. This metric set is not exhaustive, and some metrics complement each other. However, it shows how to cover as much as possible with a small number of metrics from a humor theory point of view, by only selecting a metric from each category and making sure all five dimensions are covered. In GOOFER, this metrics knowledge base is extensible, allowing it to improve the performance for particular type of joke and for testing new humor theories.

Framework Flow

The GOOFER framework uses the previously introduced classification and regression schemas and metric set to learn generating jokes based on a corpus of human-rated jokes. It first extracts the templates from the given jokes and transforms the dataset to the template values with their ratings for each template. This transformed dataset is used to learn classification schemas for each discovered template. A generator then proposes a large number of template values. The classification schema picks the template values that it considers best, based on its learned humor knowledge. These template values are then inserted into their template to create a set of output jokes.

Components

Human evaluation The example jokes have to be rated for the classifier to distinguish good jokes from bad jokes. This is the only place where humans are involved in the algorithm, apart from the user of the system delivering seed words and possibly input jokes.

Template extractor Jokes often have a template, and as such, can be clustered to discover these templates (Hong and Ong 2009; Agustini and Manurung 2012). The template extractor component detects templates for the template store, and extracts the template values for the metric-based rater. It might also detect additional information about the template, such as the part of speech of certain template values.

Template store The templates found by the template extractor are stored in the template store, along with their own trained template values classifier, for its metrical schema.

Metric-based rater Template values need feature values in order to be able to be processed by machine learning algorithms. The metric-based rater achieves this task using the identified metrics set, and using every metric on all (combinations of) template values that fit its prerequisites, thus mapping template values to a set of numbers.

Values generator The metrical schema requires that a generator proposes candidate template values for jokes. This component can thus be any other template-using joke generation system. It both receives a seed and some additional instructions from the related template (e.g. part of speech) about how to generate these values. This second type of information decreases the size of the generative space, but

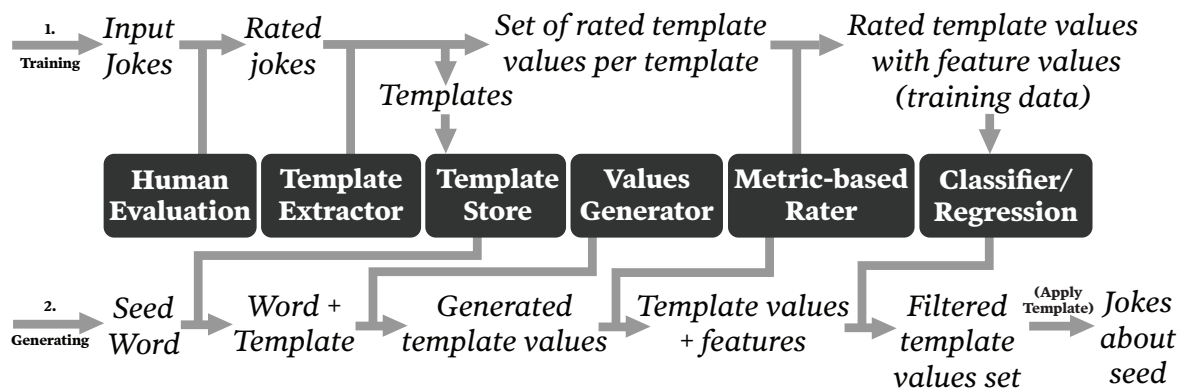


Figure 3: GOOFER framework for joke generation from examples, generalized from GAG (Winters, Nys, and De Schreye 2018).

should increase the score the classifier assigns to the generated values on average.

The generator could use any kind of text generation technique. Since we made the assumption of only having single word template values, an interesting way is using n-grams to generate related words, similar to the unsupervised analogy generator (Petrović and Matthews 2013).

As mentioned earlier, the values generator could also be another computational humor system that employs a template and schema approach, or can be shown to use an equivalent system, as we did earlier. Using such a generator in the GOOFER framework would extend an existing system by allowing it to learn from previous generations.

Classifier After the template values are converted to feature values, they receive scores from the algorithm of the classification or regression schema, which differs from template to template. A wide range of classification and regression algorithms are applicable here. In our experiments, we found RandomForests to work well for both classification as well as regression purposes (Winters, Nys, and De Schreye 2018). Additionally, when a decision tree approach is used and the knowledge base is composed of understandable humor theory metrics, the resulting decision trees might be understandable for humans. This is due to decision trees stating which attributes cause the largest impurity decreases, which can help humans learn what metrics contribute most to the funniness of specific type of joke.

Regression algorithms are useful because they are capable of dealing with real numbers. This means that they can use the average score a joke gets as the score. This is in contrast with classification algorithm that only support a certain number of classes, for which we use the mode or the rating of a single user. This can also be a disadvantage, as the perceived funniness can be entirely different between people, meaning that the average rating might often be close to the middle of the rating range.

One important factor to account for when choosing a classifier or regression algorithm, is the ability to deal with noisy metrics used in GOOFER. The framework deploys a large number of metrics on all possible combinations of the tem-

plate values, meaning many features might be noisy. The chosen algorithm should thus be resistant to this noise. Since Random Forests ignore parts of the features and of the data set, this might be one of the reasons why it performs so well in our previous experiments (Winters, Nys, and De Schreye 2018).

Code

Several components of this framework have been implemented for the evaluation. The code is available on <https://github.com/TWinters/goofer>.

Evaluation

Generalized Analogy Generator

In order to show that systems implementing the theoretical specification of the GOOFER framework work and to evaluate its results, it requires an implementation. We already implemented a subset of its components in our GAG (*Generalized Analogy Generator*) system (Winters, Nys, and De Schreye 2018), covering most GOOFER framework steps and components, apart from the template extraction step due to only having a single template (being “*I like my X like I like my Y, Z*”). The reason for this is that it simplified data collection, and because previous research on template extraction for jokes had already been successfully done (Hong and Ong 2009; Agustini and Manurung 2012).

The second simplification is the template values generator, which uses n-grams in a similar fashion as the unsupervised analogy generator (Petrović and Matthews 2013). Since the analogy joke template tends to use nouns for *X* and *Y* and adjectives as *Z*, we built our template values generator accordingly¹. As noted earlier, the POS information of template values can be found by the template extraction algorithms, implying this simplification adds no extra information. The third simplification is that the human evaluation component is also responsible for generating the input jokes

¹This is the most obvious kind of content for this template. In reality however, we noted that a significant amount of people diverge from these word types when creating this type of joke themselves, for example by naming a relation to another noun as *Z*.

used in the training data, as we build a platform to collect both. This ensured that the format of the joke is following the supported analogy joke template. This does not violate any of the assumptions of the GOOFER framework, since the input jokes were defined as coming from any source. The GAG system only used a select number of the metrics we proposed, but covered all of the five necessary properties. The GAG system thus generalized the unsupervised analogy generator using this framework.

To evaluate the GAG system, 203 different volunteers helped us collect a data set² of 524 jokes (of which 100 generated) and 9452 ratings to these jokes in total, about half of which were for evaluation. In this evaluation of GAG, we found that the best generation model was the classifier schema, with 11.41% jokes having four or more stars on five, whereas humans with similar constraints achieved this rating 21.08% of the time (Winters, Nys, and De Schreye 2018). This is a similar ratio of funniness between generated and human created jokes as the unsupervised analogy generator (Petrović and Matthews 2013), thus successfully generalising a previous system using GOOFER.

Metrics Importance Analysis

As mentioned earlier, GOOFER can find the importance of each metric for each position when using for example decision tree algorithms. The importance of each attribute for the training data using the random forest algorithm in the GAG system is given in Table 1. The importance value is based on how well a metric helps decreasing the entropy of the values in the random trees, ranging from 0 (no decrease) to 1.

Importance	Applied metric
0.67	relative_sexual_freq_2
0.67	relative_frequency_2_1
0.62	adj_vector_similarity_0_1
0.59	relative_sexual_freq_0
0.56	sexual_freq_0
0.53	word_senses_2
0.52	relative_frequency_2_0
0.52	sexual_freq_2
0.50	word_senses_0
0.50	relative_sexual_freq_1
0.50	frequency_0
0.45	frequency_2
0.43	sexual_freq_1
0.36	frequency_1
0.26	word_senses_1

Table 1: The attribute importance of every metric to a certain template value position according to the regression version of the Random Forest algorithm on the average score of the training data.

The found attribute importance seems to be conforming to the model used in the unsupervised analogy generator

²<https://github.com/twinters/jokejudger-data>

(Petrović and Matthews 2013), as their metrics roughly map to the 2nd, 3rd, 6th, 7th and 12th most important metrics. The fact that `frequency_2` is such a low scoring feature might be because of its similarity to the the top scoring feature, indicating a possible oversight for the INAPPROPRIATENESS property in that research. The found importance ranking is a good sign for GAG and GOOFER, since it seems to somewhat be able to learn these earlier found assumptions about this type of joke. The GAG system effectively generalizes the system created by Petrovic, as it eliminates the need to explicitly model the minimization function and it is capable of detecting even more possible schemas. This shows that our GOOFER framework can effectively make a more generic version of existing research.

Future Work

Generalizing Templates and Template Extraction

In this paper, we chose to work with templates that are a list of fixed strings. One issue with this is that they represent fixed sentences that do not allow small (grammatical) variations. One possible way of solving this is by defining templates using grammars with variables instead of strings and template slots alternating each other. The template extraction methods would then need to be updated, e.g. require new distance measures for grammar trees in order to find these grammar tree templates. Ideally, it would have grammar trees that could represent minimal generalizations of template values in different levels, e.g. same word, same stem, same POS, any word etc. This would increase training data per template, as the template extractor would merge training data of the similar templates.

Sentence Generation

The GOOFER framework assumes for simplicity that template values are single words. Both the discussed metrics and the generated template values have been focused for single word template values. In order to successfully generate jokes using multiple words as template values, the template value generator has to be updated to be capable of proposing such larger parts of sentences.

Probabilistic Logic Schemas

We already showed how using certain machine learning algorithms such as decision tree learners can give indication of the importance of certain attributes, giving some insight into what constitutes a good joke. We also discussed how previous work used ProLog-like notations for representing schemas. Using probabilistic logic (like ProbLog (De Raedt, Kimmig, and Toivonen 2007)) might lead to probabilistic schemas, which would combine the benefits of both the metrical schema introduced in this paper, and upgrade constraint-based schemas to the probabilistic paradigm. The rules induced by such a framework would be more human comprehensible than the algorithms we used in this paper, and could be a useful tool to create human-curated versions, leading to co-creation of joke generator specifications between humans and machines.

Conclusion

In this research, we created a computer program that is capable of learning to generate humor based on human-rated examples. We achieved this by extending and generalizing computational humor concepts such as schemas. We also used humor theory and other computational humor research in order to argue, identify and evaluate a knowledge base of humorous metrics. These findings are used in the GOOFER framework, which is capable of learning humor from rated examples. It achieves this by finding correlations between metrics applied onto template values used in templates that make a joke humorous, and indicate the most important metrics for a particular set of jokes. This framework shows how machine learning algorithms can alleviate humans from the elaborate task of crafting schemas for humor generation by hand. We thus hope that this framework can be a stepping stone towards creating a more adaptive computational sense of humor.

Acknowledgements

This work was partially supported by Research foundation - Flanders (project G.0428.15).

References

- Agustini, T., and Manurung, R. 2012. Automatic evaluation of punning riddle template extraction. In *Proceedings of the Third International Conference on Computational Creativity*, 134–139.
- Binsted, K., and Ritchie, G. 1994. An implemented model of punning riddles. *CoRR abs/cmp-lg/9406022*.
- Binsted, K.; Bergen, B.; Coulson, S.; Nijholt, A.; Stock, O.; Strapparava, C.; Ritchie, G.; Manurung, R.; Pain, H.; Waller, A.; and O'Mara, D. 2006. Computational humor. *IEEE Intelligent Systems* 21(2):59–69.
- De Raedt, L.; Kimmig, A.; and Toivonen, H. 2007. ProbLog: A probabilistic Prolog and its application in link discovery. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-2007)*, 2462–2467.
- Goldberg, K.; Roeder, T.; Gupta, D.; and Perkins, C. 2001. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval* 4(2):133–151.
- Hong, B. A., and Ong, E. 2009. Automatically extracting word relationships as templates for pun generation. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity, CALC '09*, 24–31. Stroudsburg, PA, USA: Association for Computational Linguistics.
- Kao, J. T.; Levy, R.; and Goodman, N. D. 2016. A computational model of linguistic humor in puns. *Cognitive Science* 40(5):1270–1285.
- Kiddon, C., and Brun, Y. 2011. That's what she said: Double entendre identification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, 89–94.
- Koestler, A. 1964. *The Act of Creation*. An Arkana book : psychology/psychiatry. Arkana.
- Krikmann, A. 2006. Contemporary linguistic theories of humour. *Folklore: Electronic Journal of Folklore* 33:27–58.
- Lessard, G., and Levison, M. 1992. Computational modelling of linguistic humour, tom swifties. *ALLCIACi92 Conference Abstracts* 175–178.
- Magnini, B., and Cavaglià, G. 2000. Integrating subject field codes into wordnet. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC-2000)*. Athens, Greece: European Language Resources Association (ELRA). ACL Anthology Identifier: L00-1167.
- Manurung, R.; Ritchie, G.; Pain, H.; Waller, A.; Mara, D.; and Black, R. 2008. The construction of a pun generator for language skills development. *Applied Artificial Intelligence* 22(9):841–869.
- Petrović, S., and Matthews, D. 2013. Unsupervised joke generation from big data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 228–232. Sofia, Bulgaria: Association for Computational Linguistics.
- Pilato, G.; Augello, A.; Vassallo, G.; and Gaglio, S. 2008. Ehebby: An evocative humorist chat-bot. *Mobile Information Systems* 4(3):165–181.
- Raskin, V., and Attardo, S. 1994. Non-literalness and non-bona-fide in language: An approach to formal and computational treatments of humor. *Marcelo Dascal, Pragmatics & Cognition* 2:1 31–69.
- Raskin, V. 1985. *Semantic Mechanisms of Humor*. Studies in Linguistics and Philosophy. D. Reidel, 1 edition.
- Ritchie, G. 1999. Developing the incongruity-resolution theory. *Informatics Report Series*.
- Ritchie, G. 2001. Current directions in computational humour. *Artificial Intelligence Review* 16(2):119–135.
- Ritchie, G. 2002. The structure of forced reinterpretation jokes. *Proceedings of April Fools' Day Workshop on Computational Humour (TWLT 20)* 47–56.
- Stock, O., and Strapparava, C. 2003. Hahacronym: Humorous agents for humorous acronyms. *Humor-International Journal of Humor Research* 16(3):297–314.
- Valitutti, A.; Toivonen, H.; Doucet, A.; and Toivanen, J. M. 2013. "let everything turn well in your wife": Generation of adult humor using lexical constraints. In *ACL (2)*, 243–248. The Association for Computer Linguistics.
- Venour, C. 1999. The computational generation of a class of puns. Master's thesis, Queen's University, Kingston, Ontario.
- Winters, T., and Mathewson, K. W. 2019. Automatically generating engaging presentation slide decks. In Ekárt, A.; Liapis, A.; and Castro Pena, M. L., eds., *Computational Intelligence in Music, Sound, Art and Design*, 127–141. Cham: Springer International Publishing.
- Winters, T.; Nys, V.; and De Schreye, D. 2018. Automatic joke generation: Learning humor from examples. In *Distributed, Ambient and Pervasive Interactions: Technologies and Contexts*, volume 10922 LNCS, 360–377. Streitz, Norbert.

MindMusic: Brain-Controlled Musical Improvisation

Rachel Goldstein*, Andy Vainauskas*, Margareta Ackerman*, Robert M. Keller⁺

*Department of Computer Science and Engineering

*Santa Clara University, Santa Clara, California

⁺Computer Science Department

⁺Harvey Mudd College

Abstract

In this paper, we explore a new form of creative expression through brain controlled musical improvisation. Using EEG technology and a musical improviser system, Impro-Visor (Keller 2018), MindMusic engages users in musical improvisation sessions controlled with their brainwaves. Brain-controlled musical improvisation offers a unique blend of mindfulness meditation, EEG biofeedback, and real-time music generation, and stands to assist with stress reduction and widen access to musical creativity.

Background and Motivation

Musical self-expression and meditation offer two complementary approaches to improving mental health and well-being. Musical improvisation has been shown to activate the sensorimotor and language areas of the brain that otherwise remain dormant during the performance of predetermined melodies (Lopez-Gonzalez and Limb 2012). Furthermore, brain regions that manage executive functions such as planning, abstract reasoning, and working memory were found to be deactivated, which also occurs during meditation and dreaming.

In this work, we explore a new terrain of musical creative expression by cutting out all intermediaries and controlling musical improvisation directly through brain signals. This form of expression requires no training or musical expertise, is more widely accessible than traditional musical instruments, and stands to offer a variety of mental health benefits. In particular, we utilize the lightweight EEG headband, Muse¹, to access the user's EEG data, which is subsequently used to drive musical improvisation, incorporated within the musical improvisation system, Impro-Visor.²

Real-time feedback allows the user to gauge their mental state through changes in the music, which informs the user of their mental state in regular intervals. *Biofeedback* uses instruments to provide information on one's physiological function to allow greater awareness of that function. It has been shown to be an effective way to control one's mental state (deCharms et al. 2005). By becoming aware of

one's mental state in an explicit, yet pleasantly communicated manner, the user is placed in a better position to transition to a healthier state of mind.

Direct EEG feedback is complemented through an alternate improvisation driver by allowing the user's head tilts to affect the music. The Muse device can detect head position, identifying whether the user tilts their head left or right, or keeps it centered. Head position subsequently controls the durations of notes played in the improvisation. This more easily controlled form of feedback can provide musical expression to those who cannot play musical instruments due to paralysis or other disorders.



Figure 1: Andy Vainauskas (left) and Rachel Goldstein (right) showcasing MindMusic for the Santa Clara University newspaper. Shown here wearing the Muse EEG headbands.

This paper shares early stage attempts at mind-controlled musical improvisation, whereby durations of notes in computer improvised music are effected through alpha waves and, more intentionally, head tilts. Future work will ex-

¹<https://choosemuse.com>

²<https://www.cs.hmc.edu/keller/jazz/improvisor/>

plore additional layers of interaction, as well as study the impact of this novel form of brain-computer interface (BCI) on mental health and the potential to improve access to musical self-expression for paralyzed individuals.

Previous Work

Combining EEG signals with music generation offers a variety of benefits. Alvin Lucier was one of the first to transform brainwaves into sound by amplifying his EEG signals to generate music, which offered a way for creative expression to be conducted by one's brain activity (Lutters and Koehler 2016). Loudspeakers were placed near percussive instruments to resonate when frequencies outside the human range of hearing were generated.

(Eaton, Williams, and Miranda 2015) identified that brain waves provide means of passive control in a BCMI, to allow for mental states to be approximated and mapped to relative musical phrases (Eaton, Williams, and Miranda 2015). In regards to therapeutic applications, (Keune Ne Muenssinger et al. 2010) expanded upon Brain-Computer Interfaces (BCIs) to enable creative expression for patients with ALS. Their P300-Brain Painting BCI produced meaningful experiences for both healthy and paralyzed individuals.

Effects of Mindfulness Meditation and Music on the Brain

Mindfulness meditation has been shown to reduce stress and improve mental states. When compared to relaxation meditation, mindfulness meditation can better improve focus and quiet distracting thoughts (Jain et al. 2007). Mindfulness meditation has also been shown to reduce anxiety in both a broad range of clinical patients and patients suffering from generalized social anxiety disorder (Koszycki et al. 2007). Additionally, research suggests that mindfulness-based cognitive therapy significantly reduces relapse and recurrence in patients suffering from major depression (Teasdale et al. 2000).

Music is fundamentally relaxing, especially when the listener finds the music pleasurable to listen to (Stratton and Zalanowski 1984). One study, which investigated the effects that drumming had on six soldiers combating PTSD, found a reduction in some of their symptoms after playing the instrument (Bensimon, Amir, and Wolf 2008). Another study found that not only does listening to music reduce the perception of pain, but when combined with traditional pain-management techniques, music therapy can enhance the effectiveness of pain-management for patients recovering from surgery (Bernatzky et al. 2011).

Improvisation in music has an interesting effect on the brain. A study looking at improvisation in professional jazz pianists that utilized functional MRI scans found that, when compared to well-rehearsed music, improvisation correlated to disassociated activity in the prefrontal cortex. This suggests that when engaging in improvisation, musicians are using fewer brain processes involved with self-monitoring, planning, and problem-solving and are instead using more areas of the brain associated with meditation and daydreaming (Limb and Braun 2008).

System Overview

MindMusic enables brain-driven musical improvisation. EEG signals are detected through the Muse device and a user's brain activity is then used to drive the real-time music generation. We built MindMusic within Impro-Visor (Keller 2018), a musical improvisation system by Robert M. Keller (see Figure 2).

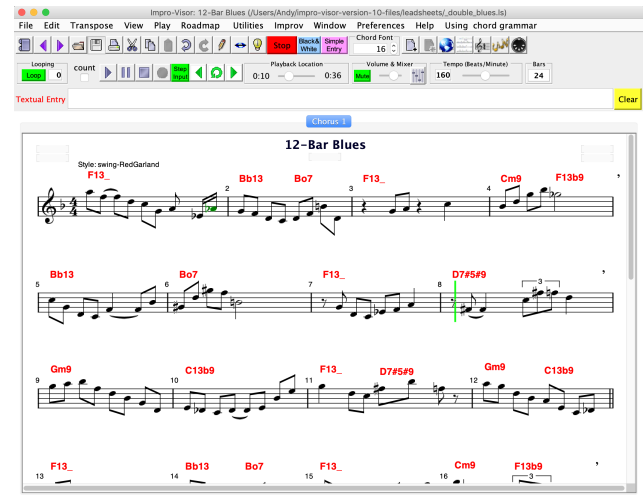


Figure 2: The Impro-Visor software displaying a lead sheet of a blues melody line.

The music created by MindMusic reflects your current brain activity, producing a musical expression based on your state of mind. By controlling the music with one's mental state, the user stands to benefit from both biofeedback and playing a musical instrument.

MindMusic comes in two modes: one for intentional musical influence and another for mindful feedback of one's brain activity (EEG). The first is a mode that enables notes to be intentionally made longer or shorter depending on which direction the user's head is tilted, allowing for the user to purposefully make adjustments as they desire.

In the head tilt mode, shorter notes are played when the user tilts their head to the left, longer notes for when the user tilts to the right, and when their head is in the middle, a mix of the two lengths are generated.

In the EEG mode, note durations reflect the current level of relaxation as measured by alpha waves, where longer durations correspond to deeper levels of relaxation.

These two modes were initially created to be used separately, however, future expansion could allow for them to be combined, with certain aspects of the music controlled with head tilts while others are reflective of brain data.

Architecture

The MindMusic architecture is shown in Figure 3. The Muse uses Bluetooth to connect to a device running the Impro-Visor software. Once Impro-Visor is initialized, an OSC server begins to listen for the data being sent from the Muse. During an improvisation session, Impro-Visor processes the

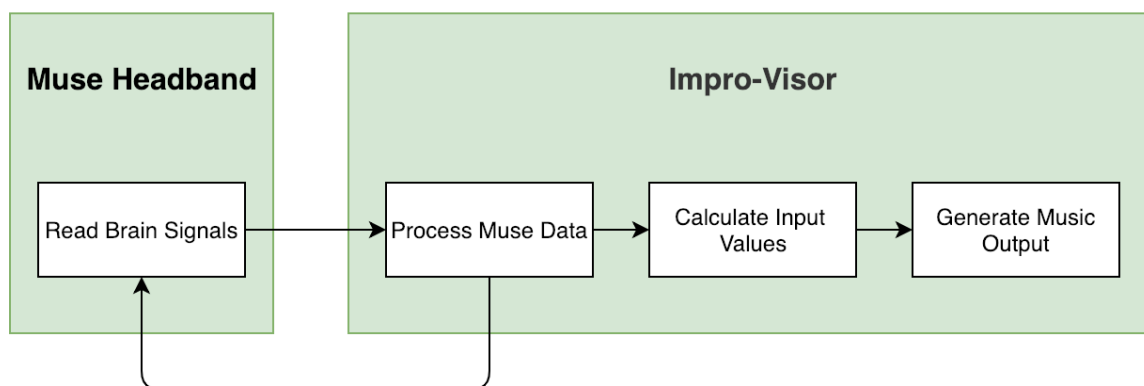


Figure 3: The MindMusic architectural diagram, which describes each individual system component.

Muse data, categorizes the current brain activity level, and generates music that is reflective of the user’s current mental state.

Muse EEG Device and Data Communication

The Muse is a headband with seven EEG sensors, two on the forehead, two behind the ears, and three reference sensors.³ It samples brainwave data at a rate of 10 Hz, and can connect to a device over Bluetooth. The Muse device offers a lightweight, affordable EEG solution for the broad consumer market.

Alpha Brainwaves

EEG brainwaves can be split into five frequency-based categories – delta, theta, alpha, beta, and gamma. At each moment, our brains have activity in each of these frequency ranges. We focus on the most prominent relaxation-related band, alpha brainwaves. There is more activity in the alpha frequency range when a person is alert but not actively concentrating or processing information. Alpha brainwaves are linked to feelings of calmness, relaxation, and tranquility while being conscious and non-drowsy.⁴ For instance, these waves tend to be more active during meditation than during normal activity. Looking at the power spectral density function (PSD), higher readings within the alpha range indicate that a user is in an “alpha state.”⁵

Brain Data Activity Categorization

To find the user’s neutral state, MindMusic begins with a 22 second calibration stage. This phase allows MindMusic to assess what constitutes a relaxed or anxious state for different users. During this period, samples are taken of the user’s alpha brainwaves. From that sampling, the user’s average alpha, μ , and standard deviation, σ , are saved.

Following the calibration stage, we calculate the average of the user’s alpha band using their most recent three seconds of alpha brainwave data. From there, the z-score of

this new sampled alpha is calculated, using the following equation,

$$z = \frac{x - \mu}{\sigma}$$

where x is the average of the most recent read values, and μ and σ are the values saved from the user’s calibration stage.

Using this z-score, these new alpha values can then be mapped to different activity states. We chose to have five different activity states: very low, low, medium, high, and very high. A z-score close to the user’s average will be closer to 0 and categorized as a medium activity level, very high and low z scores are categorized as very low and very high activity levels, and those in-between are categorized as low and high. A higher alpha value read reflects that the brain has more brainwaves that fall into the alpha frequency range, and so the user can be seen as more calm and less active.

Musical Representation

Impro-Visor uses a collection of probabilistic context-free grammars to establish a set of rules that are used for music generation (Keller 2018). Two additional grammars were created to support MindMusic’s functionality: head tilt and brainwave. These two grammars allow the user to engage in both intentional and reflective modes of musical creation, respectively.

For the head tilt grammar, if the user’s head is tilted to the left, the grammar will execute a rule that returns notes that have longer durations, such as quarter notes. If their head is tilted to the right, notes with a shorter duration are generated, such as eighth notes. When the head is held upright, a mixture of short and long notes are generated.

The brainwave grammar uses alpha waves read by the Muse that have already been categorized into an activity level. If the level is “low” or “very low”, then notes of a longer duration are generated (half or whole notes). If the level is “medium”, this means that their current activity level is similar to what was measured during the calibration stage, so quarter notes are generated. If their activity level is “high” or “very high”, then faster notes, such as eighth and sixteenth notes, are generated.

³<https://choosemuse.com/how-it-works/>

⁴<https://nhahealth.com/brainwaves-the-language/>

⁵<https://musemonitor.com>

To improve brain activity categorization, we tested MindMusic with several users to get their feedback on the accuracy of the musical representation of brain activity. Minor adjustments were made to the range calculations after discovering that achieving a “very low” or “very high” state occurred too often. Future work will include detailed user studies.

Conclusions and Future Work

We have initiated the exploration of mind-driven improvisation by effecting the note duration of a musical improvisation based on either head tilts or brain activity. There is significant potential to expand. Aspects such as pitches, volume, tempo, dynamics, note register, and note quality could also be influenced by brain data, and more advanced systems could reflect higher order concepts such as the musical tone and mood. By incorporating other waves (beta, theta, delta, gamma), a more complete picture of one’s mental state could be captured and better reflected in the music.

Mindfulness meditation has been shown to reduce symptoms of major depression and anxiety, and music has been shown to alleviate pain. Through the integration of mindfulness and music, MindMusic has the potential to lead to more effective treatment options for those suffering from similar conditions. MindMusic gives healthy individuals an audible way to connect with their current mental state, and this approach may help people with disabilities engage in musical expression when they cannot engage in this creative activity through traditional instruments.

Acknowledgments

We would like to thank Jonathan Bernal for his help throughout this project.

References

Bensimon, M.; Amir, D.; and Wolf, Y. 2008. Drumming through trauma: Music therapy with post-traumatic soldiers. *The Arts in Psychotherapy* 35(1):34 – 48.

Bernatzky, G.; Presch, M.; Anderson, M.; and Panksepp, J. 2011. Emotional foundations of music as a non-pharmacological pain management tool in modern medicine. *Neuroscience Biobehavioral Reviews* 35(9):1989 – 1999. *Pioneering Research in Affective Neuroscience: Celebrating the Work of Dr. Jaak Panksepp.*

deCharms, R. C.; Maeda, F.; Glover, G. H.; Ludlow, D.; Pauly, J. M.; Soneji, D.; Gabrieli, J. D. E.; and Mackey, S. C. 2005. Control over brain activation and pain learned by using real-time functional mri. *Proceedings of the National Academy of Sciences* 102(51):18626–18631.

Eaton, J.; Williams, D.; and Miranda, E. 2015. The space between us: Evaluating a multi-user affective brain-computer music interface. *Brain-Computer Interfaces* 2(2-3):103–116.

Jain, S.; Shapiro, S. L.; Swanick, S.; Roesch, S. C.; Mills, P. J.; Bell, I.; and Schwartz, G. E. R. 2007. A randomized controlled trial of mindfulness meditation versus relaxation training: Effects on distress, positive states of mind,

rumination, and distraction. *Annals of Behavioral Medicine* 33(1):11–21.

Keller, R. M. 2018. Impro-Visor: Jazz Improvisation Advisor for the Improviser. <https://www.cs.hmc.edu/keller/jazz/improvisor/>.

Keune Ne Muenssinger, J.; Halder, S.; Kleih, S.; Furdea, A.; Raco, V.; Hsle, A.; and Kbler, A. 2010. Brain painting: First evaluation of a new braincomputer interface application with als-patients and healthy volunteers. *Frontiers in neuroscience* 4:182.

Koszycki, D.; Bengler, M.; Shlik, J.; and Bradwejn, J. 2007. Randomized trial of a meditation-based stress reduction program and cognitive behavior therapy in generalized social anxiety disorder. *Behaviour Research and Therapy* 45(10):2518 – 2526.

Limb, C. J., and Braun, A. R. 2008. Neural substrates of spontaneous musical performance: An fmri study of jazz improvisation. *PLOS ONE* 3(2):1–9.

Lopez-Gonzalez, M., and Limb, C. 2012. Musical creativity and the brain. *Cerebrum*.

Lutters, B., and Koehler, P. J. 2016. Brainwaves in concert: the 20th century sonification of the electroencephalogram. *Brain* 139(10):2809–2814.

Stratton, V. N., and Zalanowski, A. H. 1984. The Relationship Between Music, Degree of Liking, and Self-Reported Relaxation. *Journal of Music Therapy* 21(4):184–192.

Teasdale, J. D.; Williams, J. M. G.; Williams, J. M. G.; Soulsby, J. M.; and Lau, M. A. 2000. Prevention of relapse/recurrence in major depression by mindfulness-based cognitive therapy. *Journal of Consulting and Clinical Psychology* 68(4):615–623.

Situated Novelty in Computational Creativity Studies

Marija Majda Perišić

Faculty of Mechanical Engineering
and Naval Architecture,
University of Zagreb, Croatia
mperisic@fsb.hr

Mario Štorga

Faculty of Mechanical Engineering
and Naval Architecture,
University of Zagreb, Croatia
Luleå University of Technology,
Sweden
mario.storga@fsb.hr

John Gero

University of North Carolina at
Charlotte, USA
George Mason University, USA
john@johngero.com

Abstract

This paper furthers the study of creative design by taking a situated view of novelty. A set of computational experiments is performed utilizing an agent-based model of a design team, and resulting data is used to examine the influence of a change in a situation (or a design frame) on the perception of a design's novelty in terms of its difference from existing or possible designs. The experiments demonstrate that, over the course of designing, solutions which were regarded as novel, can become not novel. They also show that a solution which was not seen as novel in one situation can be assessed as novel when a situation changes. The results, therefore, emphasize the importance of studying novelty as a situated measure.

Introduction

Design has long been recognized as a situated act (Gero 1998; Gero and Kannengiesser 2004; Suwa, Gero, and Purcell 2000). Empirical findings (Suwa, Gero, and Purcell, 2000) suggest that throughout designing, designers use their past experiences and expectations to develop interpretations of their tasks. To further the studies on creativity, Kelly and Gero (2015) developed the notion of situated interpretation and used it for studies on framing in creative tasks. Their work builds on Boden's (1996; 2004) theory of creativity and emphasizes that to understand the creative aspects of design activity it is essential to understand how the conceptual space changes. In their later work, Kelly and Gero (2017) elaborated the notion of situated interpretation and build on it to develop a paradigm of generate and situated transformation, where a situated transformation is defined as a process that draws on previous experiences to create a new design space or to modify the existing one.

To further the studies on the relationship between situatedness and creativity, this research utilized computational experiments to explore how novelty – a key aspect of creativity – is influenced by the situated changes in the conceptual space in which the design occurs.

Novelty

Novelty, its definition, assessment or reinforcement, constitutes an inevitable part of every study on creativity. Earlier work on creativity (e.g., Besemer 2006; Boden 1996) viewed novelty as a term covering aspects of originality and surprise. However, recently researchers (e.g., Maher, Brady, and Fisher 2013) argued that novelty, as a measure of difference of a design relative to the set of existing designs, does not necessarily imply violation of expectations (i.e., surprise) in a space of projected designs. Following this distinction between surprise and novelty, Maher and Fisher (2012) and Grace et al. (2015) proposed measures of novelty and surprise based on k-means clustering. Their approach includes representing each design with a set of features determining its position within the conceptual space. The measure is particularly important for computational studies of creativity as it relies on well-known and easy-to-implement mechanisms. Many additional methods for novelty detection and measurement can be found in, for example, the field of signal processing (for a review, see Pimentel et al. 2014); while for an overview of measures used in design one can consult the work of Ranjan, Siddharth, and Chakrabarti (2018).

Hypothesis

Following this brief theoretical background and empirical findings, and building on previous computational studies of novelty, this work studies novelty in design through the lens of situatedness. The notion that situational change can introduce differences in novelty assessment can be detailed through the formulation of two hypotheses:

H1: Over the time course of designing, (some of) the solutions that were previously recognized as novel, will become not novel.

H2: Over the time course of designing, (some of) the solutions that were previously not regarded as novel, can be recognized as novel.

These hypotheses are tested through a series of experiments conducted by utilizing a computational model of a design team.

Model

Within a computational model used in this study, a design team is represented as a set of cognitively rich, social agents, where each agent portrays an individual designer (Perišić et al. 2017; 2018; 2019). An agent's mental model consists of three layers: the function layer, the behavior layer, and the structure layer. At each layer, the set of nodes (of the corresponding type) represents functions, behaviors or structures known to the agent. Links between functions and behaviors, and behaviors and structures represent the associative relationships between elements known to the agent. There are no links among nodes of the same type (Gero and Kannengiesser 2004).

An agent's reasoning mechanisms are based on cognitive theories: dual system theory (Kahneman 2011) and a theory distinguishing reflexive, reactive and reflective modes of reasoning (Maher and Gero 2002). As described in (Perišić et al. 2017; 2018; 2019), each structure node is associated with a network, while each behavior node represents a range of one network property (e.g. a behavior node may represent having a clustering coefficient between 0.1 and 0.2). In this manner, one can derive structure's behavior by calculating the respective network's properties. When an agent is faced with a task (a set of requirements – i.e., required functions or behaviors), an activation impulse is generated in the function node which is deemed as relevant and passed through the links. If a structure node becomes sufficiently activated, the associated network is analyzed (i.e., its properties are calculated) and the behaviors obtained are compared to the required and expected ones. If a mismatch from the expectations is encountered, an activation impulse is sent to the function nodes relevant for the unmet requirements.

To enable agents to expand the structure space, two mechanisms were implemented: *union* – which can occur if two structures are simultaneously sufficiently active and it consists of overlaying their respective networks; and *contraction* - in which two network nodes (i.e., nodes within the structure node's network) can be collapsed into one (thus creating a new network node). Additionally, agents can learn through communication with others. If a structure node is sufficiently active and determined to meet the requirements, an agent can propose it as a solution to other agents, which in turn learn it, evaluate it in against their mental models, and rate its suitability for the task. Similarly, if two nodes are sufficiently active and the link between them is of sufficient weight, an agent can decide to communicate this link to others which can then learn the link and use it in subsequent reasoning. Through structure space expansion and processes of learning, grounding and forgetting of links, the agent's mental model develops and shapes its reasoning.

Further details on the model's implementation and performance can be found in previous work (Perišić et al. 2017;

2018; 2019). For the present study, however, one modification of the prior implementation was made. As initially modeled, a simulation was considered over when the agents found and agreed upon a single structure that satisfied the requirements. The scope of the present study requires continuous exploration and extension of the solution space. Therefore, the mechanism to avoid team fixation on a single feasible structure was implemented as follows: if a unique structure has been proposed repeatedly over the course of 10 simulation steps, and the links from relevant (i.e., required) behavior nodes to the structure are well-grounded (i.e., more than 98% of maximal link weight) in the mental models of every agent, then the structure is inhibited and the weight of relevant behavior–structure links is reduced in each of the agents' mental models. Although this results in the structure not being used in (at least some) subsequent steps, the activation level of behaviors connected to the structure remains unchanged, therefore influencing the further search. This mechanism corresponds to the situation where members of a design team all agree upon one solution and “leave it aside” to produce additional ideas, while still remembering the behavior and properties of the solution.

Design of the Experiments

For each simulation run, a task is represented as a set of network properties which a structure (i.e. structure's respective network) has to meet to be considered as a solution (i.e., to be found *useful*). To enable comparison and novelty assessment of structures, each structure is characterized by its respective network's properties. To avoid high correlations among structures' properties due to task requirements, in the present work the tasks pose requirements on the properties of the largest connected component of a structure network, while the structure's properties were calculated on the whole respective network. Each structure's characterizations consist of three values: network degree centrality, clustering coefficient and hierarchy value, based on the measure for undirected networks defined by Mones, Vicsek, and Vicsek (2012). Tasks were defined as combinations of requirements regarding diameter, closeness and/or betweenness centralities of the structure network's largest connected component.

Throughout a simulation, details on all of the agent-generated structures were collected. Further, at each time step, a reachable structure space was calculated. Reachable structure space at a time step t (RSS_t) is defined as a space of all structures which can be created by agents in a time step $t+1$. In other words, it is a space of all structures derived from the structures known to agents at the step t by utilizing union and contraction mechanisms described in the previous section. A subset of a reachable structure space consists of all reachable feasible solutions ($RFSS_t$), i.e., a space of all structures meeting the requirements which can be generated in the next step. Finally, a subset of all reachable feasible solutions can be considered as novel, thus constituting a space denoted as $RNSS_t$. To derive $RNSS_t$ from its corresponding $RFSS_t$, Mahalanobis distance was used. Mahalanobis distance is a measure frequently used for detection of outliers in multivariate data and has often been utilized in

machine learning systems to identify data distinct from the samples used for system’s training (Pimentel et al. 2014).

Overall, 300 simulation experiments were run, each terminating when the size of the RSS reached 100,000 nodes. The average sizes and standard deviations of the size of reachable structure space (RSS) and reachable feasible structure space (RFSS) over time are shown in Figure 1.

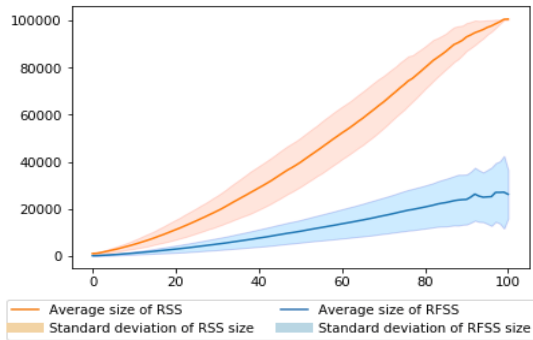


Figure 1. Average size and standard deviations of RSS and RFSS over time

Results

To test for the hypotheses posed in this paper, the number of novel structures which, over the course of the simulation, turned not novel, was counted. Similarly, the statistics include the number of structures which were initially (i.e., at the time of their first occurrence within reachable structure space) not marked as novel, only to be labeled as novel as the simulation progressed. The respective average (aggregate) numbers and standard deviations are presented in Table 1. Finally, to illustrate the dynamics of the simulated experiments and to provide deeper insights in the obtained results, a series of snapshots for one simulation experiment are extracted and presented in Figure 2. In the figure, feasible solutions are shown, and further differentiated based on their novelty status. Structures more than two standard deviations apart from the sample mean were marked as novel.

Number of Novel -> Not Novel nodes per simulation		Number of Not Novel -> Novel nodes per simulation	
Average	Standard deviation	Average	Standard deviation
187	167.06	653	256.88

Table 1. Statistics on the number of novel solutions turned not novel and not novel solutions turned novel

Discussion

As shown in Figure 1, the mechanisms which agents use to extend the solution space enable them to create new – feasible or not – structures over time. The number of novel solutions found in each simulation step increases over the course of the simulation. However, the percentage of feasible solutions which is recognized as novel slowly declines over time. More precisely, at the early stages of simulation (first

10%) when the number of novel solutions is small (less than 100), the percentage of novel solutions shows several increases and decreases, and a large standard deviation. This reflects the differences between simulation runs. Namely, while in some cases agents needed more steps to find novel solutions, at other runs some of the structures generated at the early stages of simulation were already significantly different from others. As the simulation progresses, the percentage stabilizes and starts to decline.

Interestingly, significantly more structures turned from not novel to novel, than the other way around. To explain this, one may take a closer look at the dynamics of the simulation depicted in Figure 2. At the simulation start, several nodes on the left side of the space are marked as novel. However, as the agents learn and create more structures, the space of reachable feasible solutions changes to include structures closer to those previously regarded as new. As a consequence, the notion of what is novel (i.e., different from others) gradually shifts from the left to the far right of the space (third subfigure). As the process continues, the RFSS grows, and the majority of reachable structures concentrate in a cluster on the left of the space. As a consequence, the standard deviation of the population decreases, and a (relatively) large number of structures which are outside of the cluster but were previously not regarded as novel, now become assessed as such. This example serves to show how agent-produced solutions, after some time, start to converge to “similar looking” structures satisfying the requirements. Such a process is a consequence of two factors: first, the novelty detection algorithm was not implemented in the agents themselves, therefore restricting them from detecting that subsequently generated structures moved the space towards increasingly similar structures. Secondly, the synthesis mechanisms (union and contraction, among which union is more frequently applied) led to the generation of larger, well-connected networks for which calculated measures differ to a lesser extent. Due to these characteristics (arguably, limitations) of the system, over time, the majority of reachable structures “concentrates”. This means that the number of structures which will be marked as novel and then changed to not novel decreases over time. However, as the simulation progresses, the large proportion of the previously not novel structures will become assessed as novel.

Despite the limitation that agents do not assess novelty, the results enable interesting insights. If a task is reframed to broaden the solution space, the difference in characteristics between the initial and newly obtained space may cause some solutions to no longer be regarded as novel. Perhaps more surprisingly, broadening the solution space can also cause some (previously uninteresting) solutions to stand out. This effect may likewise be discussed through the notion of “norms” and development of immutable expectations.

Finally, it would be interesting to observe whether the similar results would be obtained if the agents were modelled as the robot with real-time novelty-detection mechanism implemented in (Marsland, Nehmzow, and Shapiro 2000). Marsland et al. (2000)’s robot utilizes habituation and recovery mechanisms enabling it to get accustomed to

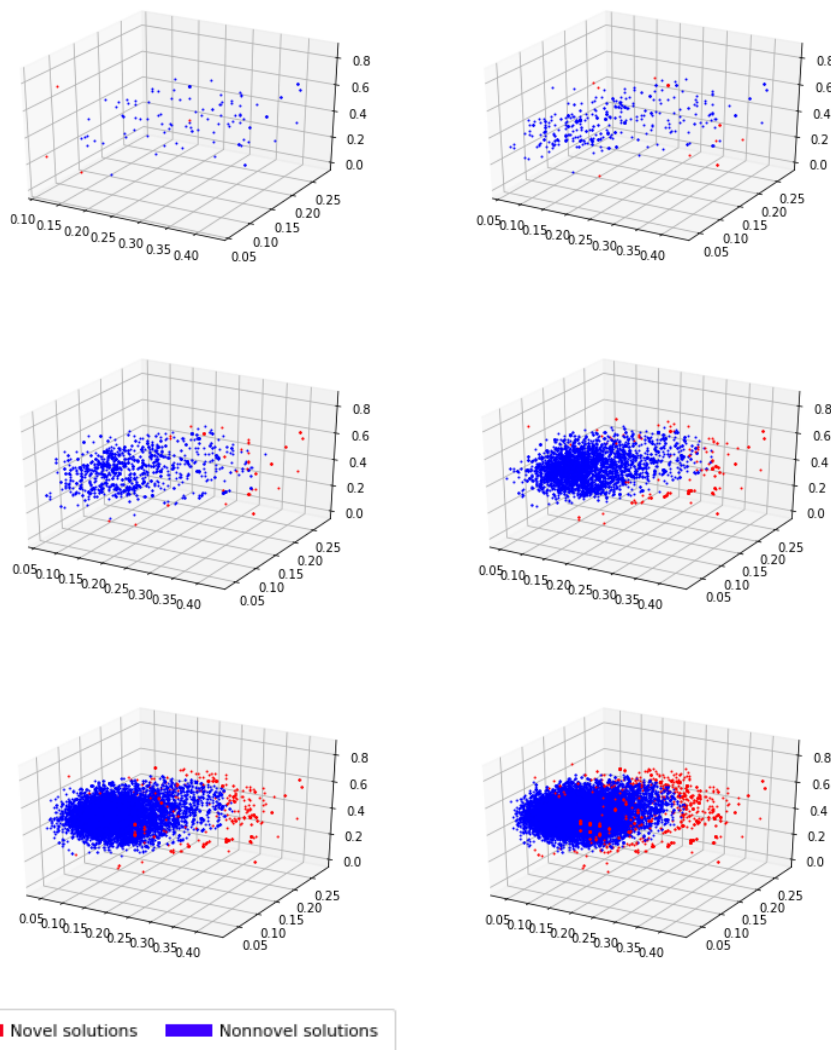


Figure 2. An example of the simulation run

stimuli, develop different value systems and forget, which results in it being able to mark the stimuli as novel even though the stimuli has already been encountered in previous stages of the simulation.

Conclusion

This work builds on the notion of situatedness and uses it to study creative aspects of design. Relying on empirical findings and computational models of creativity in design, this work explores how a vital part of creativity – novelty – changes with respect to the situation. A series of computational experiments demonstrated the importance of a frame within which the design occurs in assessing novelty. As a design frame (or a situation as defined by Kelly and Gero 2017) is changing to broaden the solution space, a design may turn from appearing as novel to being regarded as not different from the majority of others. In contrast, a change of a design frame may cause a design previously marked as

‘typical’ (i.e., not sufficiently different from other solutions), to be seen as interesting and different from others.

The simulations utilized the Mahalanobis distance to detect structures distant from the general distribution of designs. In future, experiments using different measures could be compared and assessed based on their capability to capture the notion of novelty, therefore determining whether the observed trends are emerging from the chosen novelty measure. Additionally, one may note that the current framework does not enable the dynamic introduction of new variables along which solutions can differ. It is likely that some of the subsequently added structures that were labeled as not novel would be found as novel based on some additional dimension (i.e., different network characteristic). In the present study, difference in designs is manifested through either new values for a certain attribute or as a novel combination of attributes. But, as Gero (1990) postulated and Maher and Fisher (2012) demonstrated with the example of Bloom Laptop design, new (and surprising) designs can introduce new variables (i.e., dimensions) along which designs can differ.

Nevertheless, this work demonstrates the importance of regarding novelty as a situated measure. Numerous examples from fashion (Bianchi 2002), or even the healthcare industry (Janssen, Stoopendaal and Putters 2015), show how “old” designs can again come under the spotlight due to the change in their contextual factors. In future studies, an approach similar to the one taken here can be applied to study how assessment of another creativity aspect – surprise – is influenced by situational changes.

Acknowledgements

This work is funded by Ministry of Science, Education and Sports of the Republic of Croatia, and Croatian Science Foundation project IP-2018-01-7269: Team Adaptability for Innovation-Oriented Product Development - TAIDE (<http://www.taide.org>), and has been supported in part by the National Science Foundation under grant numbers CMMI-1400466 and CMMI-1762415 to John Gero. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of National Science Foundation.

References

- Besemer, S. P. 2006. *Creating products in the age of design: How to improve your new product ideas!* Stillwater, OK: New Forums Press.
- Bianchi, M. 2002. Novelty, preferences, and fashion: when goods are unsettling. *Journal of Economic Behavior & Organization* 47(1):1–18.
- Boden, M. A. 1996. What is creativity? In Boden, M. A., ed., *Dimensions of Creativity*. Cambridge, MA / London: MIT Press. 75–117.
- Gero, J. S. 1990. Design prototypes: a knowledge representation schema for design, *AI Magazine* 11(4): 26–36
- Gero, J. S. 1998. Conceptual designing as a sequence of situated acts. In *Artificial intelligence in structural engineering*. Berlin: Springer. 165–177.
- Gero, J. S., and Kannengiesser, U. 2004. The situated function–behaviour–structure framework. *Design Studies* 25(4): 373–391.
- Grace, K.; Maher, M. L.; Fisher, D.; and Brady, K. 2015. Data-intensive evaluation of design creativity using novelty, value, and surprise. *International Journal of Design Creativity and Innovation* 3(3-4):125–147.
- Janssen, M.; Stoopendaal, A.; and Putters, K. 2015. Situated novelty: Introducing a process perspective on the study of innovation. *Research policy* 44(10):1974–1984.
- Kahneman, D. 2011. *Thinking, fast and slow*. New York: Farrar, Straus and Giroux.
- Kelly, N., and Gero, J. S. 2015. Situated interpretation in computational creativity. *Knowledge-Based Systems* 80:48–57.
- Kelly, N., and Gero, J. S. 2017. Generate and situated transformation as a paradigm for models of computational creativity. *International Journal of Design Creativity and Innovation* 5(3-4):149–167.
- Maher, M. L., and Gero, J. S. 2002. Agent models of 3d virtual worlds. In *Proceedings of the 2002 Annual Conference of the Association for Computer-Aided Design in Architecture*, 127–138. Pamaona: Association for Computer-Aided Design in Architecture.
- Maher, M. L.; Brady, K.; and Fisher, D. H. 2013. Computational models of surprise in evaluating creative design. In *Proceedings of the Fourth International Conference on Computational Creativity*, 147–151. Sydney, Australia: University of Sydney.
- Maher, M. L., and Fisher, D. H. 2012. Using AI to evaluate creative designs. In *Proceedings of the 2nd International Conference on Design Creativity*, 45–54. Glasgow, UK: The Design Society.
- Marsland, S. R.; Nehmzow, U.; and Shapiro, J. L. 2000. A real-time novelty detector for a mobile robot. In *Proceedings of European Advanced Robotics Systems Masterclass and Conference*.
- Mones, E.; Vicsek, L.; and Vicsek, T. 2012. Hierarchy measure for complex networks. *PloS one* 7(3):e33799.
- Perišić, M. M.; Štorga, M.; and Gero, J. 2017. Building a computational laboratory for the study of team behaviour in product development. In *Proceedings of the 21st International Conference on Engineering Design (ICED 17)*, 189–198. Vancouver, Canada: The Design Society.
- Perišić, M. M.; Štorga, M.; and Gero, J. 2018. Exploring the Effect of Experience on Team Behavior: A Computational Approach. In Gero, J.S. ed., *Design Computing and Cognition '18 (DCC'18)*, 595–612. Cham: Springer.
- Perišić, M. M.; Martinec, T.; Štorga, M.; and Gero, J. 2019. A computational study of the effect of experience on problem/solution space exploration in teams. In *Proceedings of the 22nd International Conference on Engineering Design (ICED 19)*. Delft, Netherlands: The Design Society.
- Pimentel, M. A.; Clifton, D. A.; Clifton, L.; and Tarassenko, L. 2014. A review of novelty detection. *Signal Processing* 99:215–249.
- Ranjan, B.; Siddharth, L.; and Chakrabarti, A. 2018. A systematic approach to assessing novelty, requirement satisfaction, and creativity. *AI EDAM* 32(4):390–414.
- Suwa, M.; Gero, J. S.; and Purcell, T. 2000. Unexpected discoveries and S-invention of design requirements: important vehicles for a design process. *Design Studies* 21(6):539–567.

Going Into Greater Depth in the Quest for Hidden Frames

João Gonçalves^{*1}, Aaron Bembek², Pedro Martins¹, and Amílcar Cardoso¹

¹CISUC, Department of Informatics Engineering, University of Coimbra, Portugal

²Harvard University, Cambridge, Massachusetts, USA

{jgconc@dei.uc.pt, bembek@g.harvard.edu, pjmm@dei.uc.pt, amilcar@dei.uc.pt}

Abstract

Semantic frames are a fundamental ingredient in computational implementations of Conceptual Blending (CB) theory. Therefore, we may ask the question of how to build them or where to retrieve them. This paper offers a solution which is to explore large-semantic networks for repeating structures resembling frames. We also include a feature the frames could have to give them a sense of completeness. Potential patterns were searched with a Multi-Objective-Evolutionary-Algorithm (MOEA) giving wider and ampler results when compared with a Single-Objective stochastic search.

Introduction

The Conceptual Blending (CB) framework (Fauconnier and Turner 2002) was suggested as a cognitive theory interpreting how conceptual integration processes occur in the human mind, as well as the creation of meaning, argumentation and the transmission of ideas (Coulson 2006). Although not devised by its authors to explain the formation of creative constructs, CB has been successfully used as the main engine in many Computational Creativity (CC) systems such as (Pereira 2005; Gonçalves, Martins, and Cardoso 2017; Cunha et al. 2017; Eppe et al. 2018; Martins et al. 2019).

The theory involves interactions between four mental spaces: two *input spaces*, a *generic space* and the *blend space* (Fig. 1). The latter contains the “output” of the CB process. Each mental space corresponds to a partial and temporary structure of knowledge built for the purpose of local understanding and action (Fauconnier 1994). In some implementations of CB, including ours, the mental spaces are stored as semantic graphs. These are networks of vertices (the concepts) connected by directed edges (the relations). Each relation/edge states a fact between a subject and an object such as:

partOf(wing, bird).

CB theory also mentions frames which are required in some computational models of CB (Pereira 2005), including one we have been working on (Gonçalves, Martins, and Cardoso 2017). They are needed to guide the blending process towards stable and recognisable wholes. Frames represent situations or interactions involving various participants and

*Corresponding author.

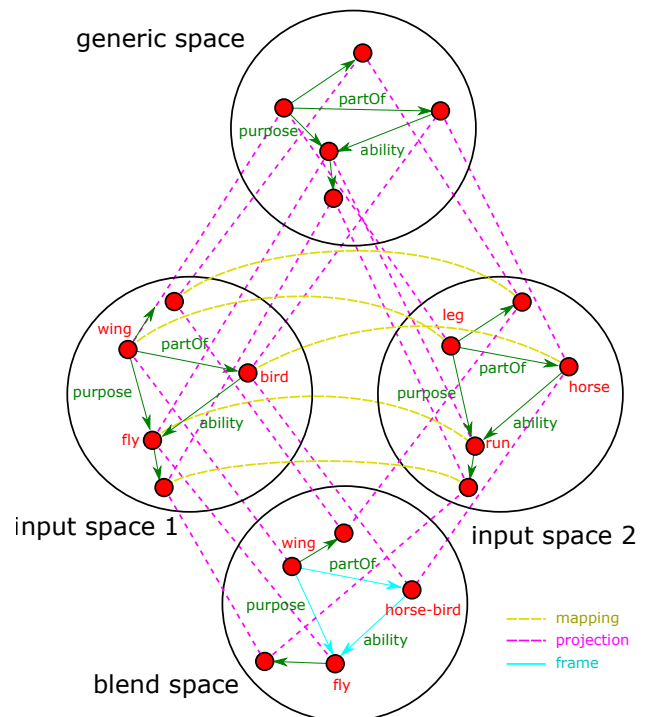


Figure 1: The four mental spaces of CB theory with two examples of input spaces: horse and bird. One possible blend of these input spaces could slightly resemble the *Pegasus*.

can be very general as well as very specific (Johnson and Fillmore 2000). Frames can be organised either as a lattice or a taxonomic structure. For example, within the domain of *motion*, the *transportation* frame provides *movers* with *means* of transportation along a *path* (Baker, Fillmore, and Lowe 1998). A more intuitive example is the frame marriage where, amongst other properties, two people have witnesses and share vows. In this case, a possible mental space containing this frame would be Mary’s marriage. In practical terms, a semantic frame is composed of either specific or abstract concepts connected by relations between them with the whole representing a meaningful entity, event or other abstract composite concept.

We now state the motivation for the current work. The first is the intention of finding useful and interesting frames on large-scale networks. The subject of finding appealing semantic frames was previously addressed with a computational model (Gonçalves, Martins, and Cardoso 2018), being the system described in this paper a direct descendant. Additionally, we were wondering if it was possible to find structures resembling frames in the same Knowledge Base (KB) functioning as the source of input spaces. Secondly, although some repositories of frames do exist - FrameNet (Ruppenhofer et al. 2016), MetaNet (David et al. 2014) and Framester (Gangemi et al. 2016), amongst others - we do think they are better aimed at linguistic systems and not easily usable by computational models of CB. Hence our second motivation, the building of a sufficiently comprehensive repository of frames to help with computational implementations of CB. The third motivation is the implementation of visual tools to help researchers who work with semantic frames in general.

After this introductory section, we follow with a short description of the CB theory, then with the importance of semantic frames in CB and our latest approach to frame mining. Later, we present and discuss the results and conclude on our findings, followed by what we expect as further work.

Conceptual Blending (CB) Theory

The input spaces serve as the initial sources of knowledge and supply the content that will be blended. A partial mapping is first established between both input spaces, reflecting a sense of similarity between them. This mapping associates the input spaces and is mirrored in the generic space, encapsulating the elements shared by the input spaces. A custom selection of this mapping is used to partially project structures from both input spaces - including nearby elements - integrating them in an emerging structure called the blend.

The integration of input elements from the input spaces (Fauconnier and Turner 1998) in the blend space is split in three sub-tasks: **composition**, **completion** and **elaboration**. The first is the projection of elements from the input spaces into the blend space. Completion corresponds to the use of existing knowledge in the form of background frames and the generation of meaningful structures in the blend. Elaboration performs cognitive work in the blend according to its ongoing emergent structures. The order of these tasks does not need to be pre-determined and several iterations may be necessary (Pereira and Cardoso 2003).

The CB process allows for substantial diversity of generated blends that - depending on the quantity of knowledge being handled - may be computationally unbounded (Martins et al. 2016). To guide the integration process towards highly integrated, coherent and easily interpreted blends, (Fauconnier and Turner 2002) proposed eight **optimality principles**. We outline two principles stating the relevance of frames according to Fauconnier and Turner:

Integration - the blend must be perceived and manipulated as a unit. Every element in the blend should have integration.

Pattern Completion - elements in the blend should be completed by using existing patterns as additional inputs. A

completing frame should be used that has compressed versions of important vital relations between the inputs.

Hunting for Semantic Frames

The frames are assumed to be patterns composed of more than one relation between three or more concepts. An example is seen in Fig. 1 as the three connected relations *purpose*, *partOf* and *ability*. The system searches for recurring patterns such as those in a KB that contains semantic graphs representing the input spaces. The concepts present in the patterns are converted to variables (words starting with a Capital letter) and a Prolog query is made from the pattern. Using the example shown in Fig. 1 this would be the following query:

ability(A, C), purpose(B, C), partOf(B, A).

A frame is satisfied if instantiating its variables with different concepts present in the KB, the frame's conditions agree with the KB's facts. The number of unique possibilities for these variables (as well as their combination) represents the prevalence of the frame's structure in the KB. We see this as an important factor and it is one of multiple objectives to be solved. The remaining objectives are explained in the following section.

Multi Objective Evolutionary Algorithm (MOEA)

The search for patterns resembling frames is handled by a MOEA evolving a set of chromosomes where each encodes a pattern. These are mutated according to the relations existing in the KB, adequately adding relations from the KB or removing existing ones from the pattern. The mutation is done while maintaining consistency between the relations in the pattern and in the KB.

In (Gonçalves, Martins, and Cardoso 2018), the fitness of each pattern was a weighted sum of various objectives. It is now a set of various competing objectives. This allows for a further exploration of the search space, covering a higher diversity of frames within the range of all the objectives. A GUI (Fig. 2) was also implemented to help with the visualisation of the Non-Dominated Set (the solutions not dominated by others).

Three objectives were outlined, all to be maximised. The first is the number of solutions a frame has. The second is the number of unique labels existing in the pattern's relations. The third is an idea we named *ucycles* (short for undirected cycles). The stochastic algorithm of the previous work rarely found patterns with *ucycles* (at most one in a million patterns) and thus the justification for an additional objective. Our reason for frames with *ucycles* is more of a subjective one, but we think that those frames have a sense of completeness, because the concepts in a *ucycle* are inter-related and closed as a whole.

A *ucycle* of a graph is a path of *distinct* edges where any vertex is reachable from itself, ignoring the direction of the edges. The number of *ucycles* a pattern contains is calculated using an adapted depth-first expansion. It goes through all the connected vertices of the pattern while skipping previously expanded relations to prevent the algorithm from

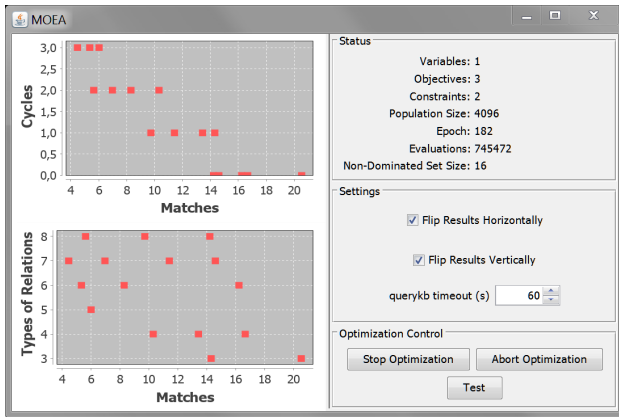


Figure 2: Graphical User Interface of the MOEA with three objectives and the Non-Dominated Set (red squares).

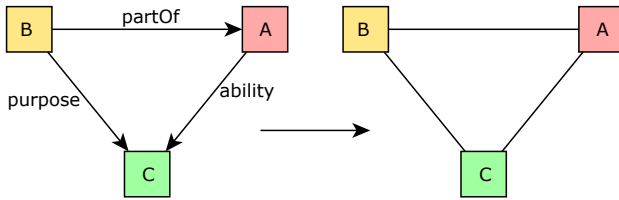


Figure 3: Example of a frame with an undirected cycle.

getting stuck. The expansion traverses all connected edges existing in the pattern, regardless of their direction (Fig. 3).

Counting the Materialising Frames

Our multithreaded *querykb* tool counts the number of solutions satisfying a possible frame, one of the objectives in our MOEA. The number of occurrences of a pattern can be immense, and many patterns occur too many times to be counted using a 64-bit integer. Instead, the count is internally represented as a Big Integer. The logarithm to the base 10 of this Big Integer is then returned to the MOEA. Since we only care about the number of solutions and not what they actually are, we can often avoid explicitly enumerating all of them. Consider the simple conjunctive query

$$p(U, V), q(X, Y).$$

A naive approach to counting the number of solutions to this query would explicitly enumerate all substitutions ranging over $U, V, X,$ and Y that make this conjunction true. However, to count the number of solutions we can just multiply the cardinality of p by the cardinality of q . Our tool generalises this intuition to handle cases where variables are shared between conjuncts. As it evaluates a conjunctive query, it maintains an intermediate relation that pairs substitutions with integers. The integer associated with a substitution represents how many substitutions have been merged into that single substitution. Substitutions are merged when they are equivalent modulo bindings for variables that do not appear later in the query. For example, when the tool evalu-

ates the first conjunct of the query

$$p(X, Y), q(Y, Z).$$

it will produce an intermediate relation with n entries, where n is the number of distinct values of Y such that $p(X, Y)$ holds for some X . Each entry associates a substitution for Y with the number of bindings for X such that $p(X, Y)$ holds for that fixed value of Y ; that is, the entry represents the result of merging together all the discovered substitutions ranging over X and Y that have that binding for Y .

To maximise the opportunities to merge substitutions, we use a query planning heuristic that exploits the structure of the graph representation of the query. In this undirected graph, vertices represent variables and edges signify that two variables appear in the same conjunct. Our heuristic searches for a bridge E whose endpoints have high degrees (a bridge is an edge whose deletion leads to a larger number of components). Call the component at one end of the bridge A and the component at the other end of the bridge B . The heuristic constrains the query evaluation plan so that all the conjuncts that appear in A are evaluated before the conjunct represented by E , and all the conjuncts that appear in B are evaluated after the conjunct represented by E . This means that by the time query evaluation reaches the conjunct represented by E , the bindings for all the variables in A (except the endpoint of E) have been merged away, as these variables cannot appear in any conjunct in B . The same heuristic is then recursively applied to the components A and B . For a component containing no bridges, we order the conjuncts using a heuristic that eagerly minimizes the domain of substitutions in intermediate relations.

Results and Discussion

The KB supplying the facts was a custom version of ConceptNet V5 (Speer and Havasi 2012) with 1 229 508 concepts and 1 791 604 relations. We removed from the KB four relations: *isa*, *derivedfrom*, *synonym* and *similar*to. In our opinion, these relations are very generic and do not seem to be fruitful for the CB process. Without these the KB had 35 types of relations.

We used the MOEA Framework (Hadka 2015) with NSGA-II (Deb et al. 2000) as the evolutionary algorithm. The population size was 4096 chromosomes per epoch. The MOEA was executed on a machine with two Intel Xeon eight-core E5-2667v2 processors and 64 GB of RAM. JVM's heapsize was set to a maximum of 48 GB. The *querykb* tool used a block size of 256, 32 threads and a processing time limit for each pattern of five minutes. Five experiments were executed averaging 48 ± 24 hours and 700 ± 200 epochs per experiment.

We accumulated all the experiments in single dataset with 90 964 patterns. Given the colossal amount of patterns we developed a graphical tool to help with both the filtering and selection of promising frames (Fig. 4). We did not find on the web a graphical tool allowing a visualisation of such a large number of graphs, which inspired us to create ours. It shows the patterns as semantic graphs, allows their sorting according to both the objectives and properties of graphs and filtering patterns within a given range of properties or

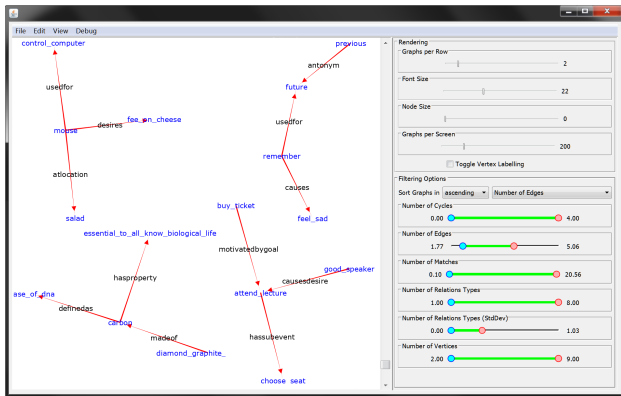


Figure 4: The semantic graph visualisation tool which also serves as a filtering aid for frames.

objectives. The tool supports the rendering of most types of semantic graphs and will be made freely available¹.

We now disclose a few patterns that we believe are interesting, including humorous ones. The patterns' variables are instantiated with examples of concepts (shown in blue) which fully satisfy the pattern in order to be easier to understand. The patterns can be seen in Figs. 5 and 6 with both figures differing on the existence of *ucycles*. The composition of the patterns is highly variable, mainly regarding the relations but we think that they can be used as representation of frames, given their recognisable structure. All these patterns had at least 1 000 occurrences in the KB.

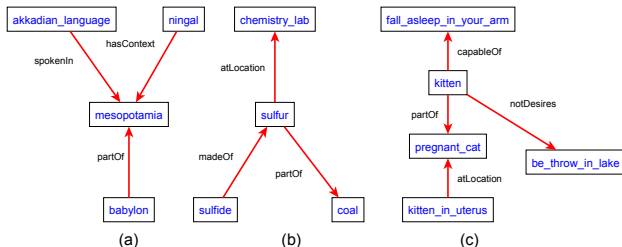


Figure 5: Example of three patterns from the experiments.

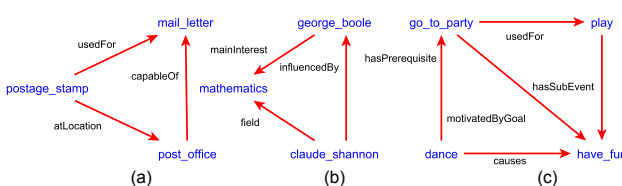


Figure 6: Three patterns with *ucycles*, the first two with one *ucycle* and the rightmost pattern with two.

The pattern in Fig. 5a may represent a frame where a place (*mesopotamia*) has a given spoken language (*akkadian language*), partitioned in sub-regions (one of which is *babylon*) with some element (the goddess *Ningal*) belonging to

¹<https://github.com/jcgon>

the region's context. Fig. 5b shows a pattern depicting an entity (*sulfur*) made of entity (*sulfide*) being part of a larger object (*coal*) found at a specific place (*chemistry lab*). Finally, given the peculiar concepts shown in Fig. 5c we leave the interpretation of this amusing pattern to the reader. We further add that some KBs (such as ConceptNet) contain erroneous, biased and funny facts (Baydin, de Mántaras, and Ontañón 2012) such as these. However, we think that in the context of CC those facts might be fruitful.

Fig. 6 contains three examples of patterns with *ucycles*. The pattern in Fig. 6a represents an entity (*postage stamp*) located somewhere (*post office*) with both the entity and the place having the same activity (*mail letter*) through different possibilities (*usedFor* and *capableOf*) Fig. 6b serves as a good case history of two entities (mathematicians *George Boole* and *Claude Shannon*) having the same *interestfield* but with one influencing the other. Hence, this frame could be classified as the “*source of inspiration*” frame. Lastly, Fig. 6c relates two activities (*dancing* and *playing*) with conditions (*go to party*) as well as outcomes (*having fun*).

Conclusions

We have presented a system designed to discover repeating patterns in large-scale semantic graphs which can be used as frames in computational models of CB. We illustrated how the system mines KBs for interesting patterns using a MOEA and a specialised tool to more efficiently compute the frequency of the patterns involved in the process. We also believe that using frames containing *ucycles* could benefit the blending process with the contribution of closely connected elements with a sense of completeness.

Further Work

We plan to create a repository with the most interesting frames found so far. We could also execute our frame finding system in other KBs such as NELL (Mitchell et al. 2015) to discover even more promising frames. Another idea to be explored is the definition of a large set of useful frames in an ontology. But above all, the KB of frames is expected to be used in a follow-up of our computational model of CB. The impact of frames in the emerging blends will be then better understood, as well as their required characteristics. Those characteristics would then be used to improve our MOEA in the search for suitable frames. We might also improve our *querykb* tool by using a more sophisticated query planning mechanism (for example, one that estimates the sizes of intermediate relations given what is known about the KB). Alternatively, instead of finding the exact number of solutions, we could try to find an approximate count, perhaps by extending the recent algorithm of (Iyer et al. 2018).

Acknowledgements

João Gonçalves is funded by Fundação para a Ciência e Tecnologia (FCT), Portugal, under the PhD grant SFRH/BD/133107/2017.

References

- Baker, C.; Fillmore, C.; and Lowe, J. 1998. The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, 86–90. Association for Computational Linguistics.
- Baydin, A. G.; de Mántaras, R. L.; and Ontañón, S. 2012. Automated generation of cross-domain analogies via evolutionary computation. *CoRR*.
- Coulson, S. 2006. Conceptual blending in thought, rhetoric, and ideology. *APPLICATIONS OF COGNITIVE LINGUISTICS* 1:187.
- Cunha, J. M.; Gonçalves, J.; Martins, P.; Machado, P.; and Cardoso, A. 2017. A pig, an angel and a cactus walk into a blender: A descriptive approach to visual blending. In *Proceedings of the Eighth International Conference on Computational Creativity (ICCC 2017)*.
- David, O.; Dodge, E.; Hong, J.; Stickles, E.; and Sweetser, E. 2014. Building the metanet metaphor repository: The natural symbiosis of metaphor analysis and construction grammar. In *The 8th International Conference on Construction Grammar (ICCG 8)*, Osnabrück, Germany.
- Deb, K.; Agrawal, S.; Pratap, A.; and Meyarivan, T. 2000. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In *International conference on parallel problem solving from nature*, 849–858. Springer.
- Eppe, M.; Maclean, E.; Confalonieri, R.; Kutz, O.; Schorlemmer, M.; Plaza, E.; and Kühnberger, K.-U. 2018. A computational framework for conceptual blending. *Artificial Intelligence* 256:105–129.
- Fauconnier, G., and Turner, M. 1998. Conceptual integration networks. *Cognitive Science* 22(2):133–187.
- Fauconnier, G., and Turner, M. 2002. *The Way We Think*. New York: Basic Books.
- Fauconnier, G. 1994. *Mental Spaces: Aspects of Meaning Construction in Natural Language*. New York: Cambridge University Press.
- Gangemi, A.; Alam, M.; Asprino, L.; Presutti, V.; and Recupero, D. R. 2016. Framester: a wide coverage linguistic linked data hub. In *European Knowledge Acquisition Workshop*, 239–254. Springer.
- Gonçalves, J.; Martins, P.; and Cardoso, A. 2017. Blend city, blendville. In *Proceedings of the Eighth International Conference on Computational Creativity*.
- Gonçalves, J.; Martins, P.; and Cardoso, A. 2018. Drilling knowledge bases for hidden frames. *Proceedings of TriCoLore 2018*.
- Hadka, D. 2015. Moea framework - a free and open source java framework for multi-objective optimization.
- Iyer, A. P.; Liu, Z.; Jin, X.; Venkataraman, S.; Braverman, V.; and Stoica, I. 2018. {ASAP}: Fast, approximate graph pattern mining at scale. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, 745–761.
- Johnson, C., and Fillmore, C. 2000. The framenet tagset for frame-semantic and syntactic coding of predicate-argument structure. In *1st Meeting of the North American Chapter of the Association for Computational Linguistics*.
- Martins, P.; Pollak, S.; Urbančič, T.; and Cardoso, A. 2016. Optimality principles in computational approaches to conceptual blending: Do we need them (at) all? In *Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2016)*. Paris, France: Sony CSL.
- Martins, P.; Oliveira, H. G.; Gonçalves, J.; Cruz, A.; Cardoso, A.; Znidarsic, M.; Lavrac, N.; Linkola, S.; Toivonen, H.; Hervás, R.; Méndez, G.; and Gervás, P. 2019. Computational creativity infrastructure for online software composition: A conceptual blending use case. *IBM Journal of Research and Development* 63(1):1–17.
- Mitchell, T.; Cohen, W.; Hruschka, E.; Talukdar, P.; Beteridge, J.; Carlson, A.; Dalvi, B.; Gardner, M.; Kisiel, B.; Krishnamurthy, J.; Lao, N.; Mazaitis, K.; Mohamed, T.; Nakashole, N.; Platanios, E.; Ritter, A.; Samadi, M.; Settles, B.; Wang, R.; Wijaya, D.; Gupta, A.; Chen, X.; Saparov, A.; Greaves, M.; and Welling, J. 2015. Never-ending learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI-15)*.
- Pereira, F. C., and Cardoso, A. 2003. Optimality principles for conceptual blending: A first computational approach. *AISB Journal* 1(4).
- Pereira, F. C. 2005. *Creativity and AI: A Conceptual Blending approach*. Ph.D. Dissertation, University of Coimbra.
- Ruppenhofer, J.; Ellsworth, M.; Petruck, M.; Johnson, C. R.; and Scheffczyk, J. 2016. *FrameNet II: Extended theory and practice*.
- Speer, R., and Havasi, C. 2012. Representing general relational knowledge in conceptnet 5. In *LREC*, 3679–3686.

Assessing Usefulness of a Visual Blending System: “Pictionary Has Used Image-making New Meaning Logic for Decades. We Don’t Need a Computational Platform to Explore the Blending Phenomena”, Do We?

João M. Cunha, Sérgio Rebelo, Pedro Martins and Penousal Machado

CISUC, Department of Informatics Engineering

University of Coimbra

{jmacunha,srebelo,pjmm,machado}@dei.uc.pt

Abstract

Visual blending can be employed for the visual representation of concepts, merging input representations to obtain new meanings. Yet, the question arises whether there is any need for a computational approach to visual blending. We address the topic using different points of view and conduct two user studies to assess the usefulness of a visual blending system.

Introduction

The mere technical possibility to implement a given computational system can be regarded as insufficient to justify its actual implementation. In most cases, there is the question of how useful a given system would be to the user – e.g. is there really a need for a computational system for meaning-making through visual blending? In fact, this question can be analysed from multiple perspectives, for example by analysing existing research work and open issues.

First, by focusing on visual blending research, we observe that there is the pursuit of a system that is able to turn concepts into visual representations. Confalonieri et al. (2015) propose argumentation as a way to evaluate and refine the quality of blended computer icons. Xiao and Linkola (2015) present a semi-automatic system to produce visual compositions for specific meanings (e.g. *Electricity is green*). Ha and Eck (2017) train a recurrent neural network that generalises concepts and can be used for representing new concepts by interpolating between several concepts. Similarly, Karimi et al. (2018) describe a deep learning approach focused on conceptual shift and present the possibility of using it to aid humans in blend production.

In order to implement a general purpose visual blending system, a large repository of visual representations with adequate format is required. However, most available repositories only provide raster images, which demand complex computer vision techniques to be used in a blending process. On the other hand, the Emoji set fulfils both requirements due to its large conceptual coverage (2823 emoji in Emoji 11.0) and appropriateness of image format – e.g. Twemoji is composed of fully scalable vector graphics, appropriate for visual blending (Cunha et al. 2017).

The Emoji’s suitability for blending is explored by Cunha et al. (2018b), who present a computational system that visually represents user-introduced concepts through visual

blending. The system is aligned with research on emoji generation (Puyat 2017; Radpour and Bheda 2017) but has a different focus – it uses emoji as a mean and not as a goal.

The second version of the system includes an interactive evolutionary engine (Cunha et al. 2019), which allows the production of solutions that match the user preference. The current version of the system – the focus of this paper – brings yet another dimension into play by giving a co-creative nature to the user-system relation.

Computational approaches to co-creativity establish a collaboration between several agents, one of which is required to be artificial. It leads to a shared creative process where agents contribute to the same goal – e.g. drawing (Davis et al. 2016) or game level design (Yannakakis, Liapis, and Alexopoulos 2014). However, examples of co-creative approaches from the visual domain mostly focus on sketching (Davis et al. 2016; Karimi et al. 2018) or abstract icons (Liapis et al. 2015), and not on pictogram generation.

Despite providing evidence of value in three different fields, the analysis in terms of research interest presented in the previous paragraphs is, in our opinion, not enough to justify such a system – providing a strong argument requires putting the system in a real-world situation. In this paper, we focus on the assessment of the usefulness of a computational system for visual representation of concepts through visual blending (Cunha, Martins, and Machado 2018b). Our main contributions are: (i) the description of two user studies and (ii) an overall discussion using a multi-perspective approach on the usefulness of visual blending systems for new meaning making.

The System

This paper focuses on the assessment of the usefulness of a computational system that uses visual blending for the representation of concepts. The system uses an interactive evolutionary approach to produce visual representations for concepts introduced by the user (Cunha, Martins, and Machado 2018b; Cunha et al. 2019). As the system itself is not the focus of this paper, we will only describe it at a general level. The system integrates data from the following online open resources: (i) Twemoji 2.3; (ii) EmojiNet (Wijeratne et al. 2017); and (iii) ConceptNet (Speer and Havasi 2012).

The current version establishes a co-creative interaction between user and system. In this interaction, solutions are

produced and the user is able to actively give feedback on their quality, leading to the evolution of better solutions. Briefly describing, the interaction could be said to have three agents: a solution generator (system), an evaluator (user) and an artificial evaluator (system). The latter agent is capable of selecting individuals based on its idea of quality and storing them in its own archive.

The solutions are produced using a process of visual blending, which consists in merging two input emoji (emoji A, the replacement, and emoji B, the base). Three different blend types can be used (Phillips and McQuarrie 2004): Juxtaposition (JUX) – two emoji are put side by side or one over the other; Replacement (REP) – emoji A replaces part of emoji B; and Fusion (FUS) – two emoji are merged together by exchanging parts.

Despite the existence of these three types of blend, in the previous versions of the system (Cunha, Martins, and Machado 2018b; Cunha et al. 2019) only juxtaposition and replacement were used. In this paper, the system tested already includes the fusion blend type.

Assessing Usefulness

As already mentioned, in this paper we delve into the usefulness of a computational system that uses visual blending of emoji to visually represent concepts. We identify three questions that we will address and aim to present evidence that points to possible answers. The three questions are: (Q1) Is visual blending effective in the visual representation of concepts? (Q2) Are all blend types equally adequate? (Q3) Is our system useful to users? In order to address the questions, we conducted two user studies. Study #1 focuses on Q1 and Q2, whereas study #2 mainly addresses Q3.

User Study #1: Visual Blending Effectiveness

The first user study is part of a larger study with single and double word concepts currently being conducted, in which preliminary results indicate that visual blending is not appropriate for one-word concepts, especially concrete ones (e.g. dog). As such, we chose to focus on two-word concepts. In spite of study #1 having several aspects that could be investigated, in the scope of this paper we mainly use it for two purposes: to investigate the effectiveness of visual blending in concept representation and to gather blends that represent a set of concepts, used in study #2 (see Fig. 1). Other aspects will be left for future work.

The study was conducted with 8 participants, who were asked to use the system to generate visual representations for a set of concepts. The concepts were selected from a list built by crossing a noun-noun compound dataset (Fares 2016) with a concreteness ratings dataset (Brysbaert, Wariner, and Kuperman 2014), which was divided into groups based on semantic concreteness and quantity of emoji retrieved by the system for each concept. For each participant, a set of five concepts was randomly built, aiming for variety and guaranteeing that each participant had at least one concept from each group. As the goal was to achieve maximum conceptual coverage, we decided to avoid concept repetition.

The participants were asked to use the system to evolve

Table 1: Results in number of occurrences of a given type of blend in exported blends for each emoji quantity group – small (≤ 5), medium (> 5 and ≤ 15) and large (≥ 25) – for juxtaposition (JUX), replacement (REP) and fusion (FUS). The “?” column refers to cases in which it was not possible to identify the type of blend and “hidden” to cases in which one of the emoji was hidden.

	JUX	REP	FUS	?	hidden
<i>small</i>	2	3	1	3	3
<i>medium</i>	8	6	0	1	2
<i>large</i>	4	9	2	0	2
	14	18	3	4	7

blends that, in their opinion, represented the concept and export the solutions which they considered the best, among the ones considered good solutions – i.e. good representations of the concept. In case no solution represented the concept, none was to be exported.

Results From a total of 40 concepts we obtained the following results: no solution was exported in 11; in 21 only one solution was exported; and in 8 more than one solution was exported. The fact that the process of visual blending was able to lead to good solutions for the majority of the concepts seems to indicate that it is a useful method for concept representation (Q1). Moreover, the results also show that the system is able to present the user with more than one good solution.

In order to further investigate the suitability of visual blending in the representation of concepts, we analysed a total of 39 blends exported by the participants in terms of blend type. The results show that juxtaposition and replacement are used in the majority of the exported blends and fusion is barely used – see Table 1 (Q2). In addition, in some cases, it was not possible to ascertain the type of blend, as one of the emoji was hidden. Another emoji hidden situation occurred in a fusion blend, in which the replacement emoji was not perceivable. We identified the cases in which one of the emoji was hidden in the blend.

User Study #2: Usefulness in Real World

The capability of a user to find a solution that visually represents a given concept with the system does not actually present much evidence of the usefulness of the system itself. For this reason, we conducted a study with the goal of comparing creative production by the user alone with results obtained with the system and assess the perception of quality by the user.

As such, we used the blends exported by the participants of study #1 – these were considered as good visual representations. From all blends exported, we selected only one per concept, using the ones identified as the best when more than one had been exported. Then we excluded the ones in which one of the emoji was being hidden, as these could not be considered as visual blends. This resulted in a set of 22 concepts and corresponding visual representations (see Fig. 1). The set was divided into three groups, balanced in



Figure 1: Blends obtained in study #1 and used in study #2

terms of semantic concreteness. Then each group was given to participants and each concept was tested with a minimum of 15 participants. Due to participant availability, the first group of concepts was tested with 15 participants, the second with 19 and the third with 22. In total, 56 users with ages between 19 and 27 (*average* = 20.4 and *standard deviation* = 1.6) participated in the study, all with background in graphic design. Each participant received a list of concepts and had to complete a survey for each concept. The survey was divided into two parts and was composed of five tasks. First, the participant was asked to conduct four tasks for each concept: **T1** Do you understand the concept? **T2** Draw the concept. **T3** Describe the drawing in few words. **T4** How well does the drawing represent the concept?

Tasks T1 and T4 required the participant to use a scale from 1 (not at all) to 5 (perfectly). In case the participant did not understand the concept (T1), the remaining tasks should be ignored. The participants were told to use a quick drawing style, similar to the one used in games such as “Pictionary”. After conducting the four tasks for every concept, the generated blends of each concept were shown and the participant was asked to answer the following question for each concept, using the previously describe 1-5 scale: (T5) How well does the blend represent the concept?

Results The study resulted in a total of 414 concept tests – group 1 was composed of 7 concepts and was tested with 15 participants; group 2 had 7 concepts and was tested with 19; and group 3 had 8 concepts and was tested with 22. From the total of tests, 76 had to be excluded from the study due to invalid answering: in 3 no answer was given to any of the tasks; in 24 no answer was given regarding the familiarity with the concept (T1); in 40 the quality of the blend was not evaluated (T5); and in 9 a visual representation was drawn but not evaluated (T4).

In addition to these validity exclusions, for our analysis we only considered tests in which the participant reported to know the concept well or perfectly ($T1 \geq 4$). We are aware that this procedure reduced the number of answers consid-

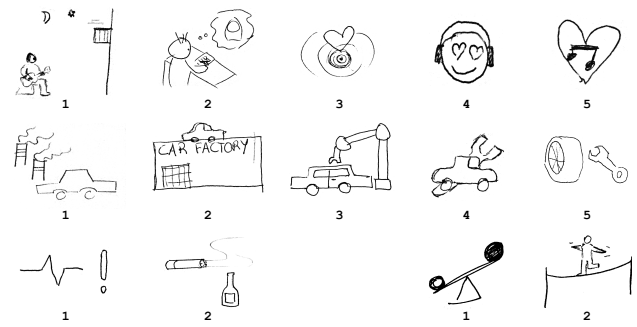


Figure 2: User drawings obtained in study #2 for *love song* (top row), *car factory* (middle row), *health risk* (bottom row, left side) and *balancing act* (bottom row, right side)

erably but it would make no sense to analyse tests in which the concept was not known to the participant – such would invalidate the results. It is important to notice the notable difference between valid tests (V in Table 2) and valid tests in which the user understood the concept (A in Table 2). This may be due to two factors: concept complexity and participant language difficulties (the participants were not native English speakers).

The results of the comparison between blend and drawing shown in Table 2 only consider the valid tests in which the user understood the concept (A). The cases in which no drawing was conducted are included in the value of better blend. Comparing the results from T4 (drawing) and T5 (blend) allow us to assess usefulness of the system.

In 8 out of the 9 concepts with high understanding rates – good understanding in the majority of the valid tests – the blend was considered better than the drawing by the majority of the participants (underlined in Table 2) and this majority was even absolute in 4 from these concepts. From the remaining 14 concepts, in 4 the blend was considered better by the majority of the participants, and in 2 the results obtained by the blend and the drawing were equal. Moreover, in two cases, the participant, despite knowing the concept, was not able to draw it and evaluated the blend as equal or better than good ($T5 \geq 4$). In contrast, the drawing was only better than the blend for the absolute majority of A in five concepts, four from which had very low understanding of the concept (less than 27% had $T1 \geq 4$). These results indicate that the system would be helpful to the user in 14 of the concepts, and its usefulness is particularly obvious in 8 from these concepts (36% of the 22).

When analysing the drawings made by the users, it is easy to observe how complex some of them are – e.g. drawings 1, 2 and 3 for *love song* in Fig. 2 were described by the users as “serenade”, “writing a love song” and “sound waves”. Yet, it is questionable whether these drawings are perceived as *love song*. Moreover, some drawings could even be more closely related to other concepts – for *car factory*, 1 could be perceived as a driving car, and 4 and 5 could be interpreted as the icon for a garage for fixing cars. In example 2, the user even included the label “car factory” to make it perceivable. In most of these examples, the blend obtained better quality

Table 2: Results of study #2 for each concept – number of tests conducted (T); mode (*mo*) and median (\tilde{x}) for the tasks T1 (concept understanding), T4 (drawing quality) and T5 (blend quality); number of valid tests (V); number of tests analysed (A); percentage of A in which the blend was worse (B < D) and better (B > D) than the drawing (includes absence of drawing).

	T	T1		T4		T5		V	A	B < D	B > D		T	T1		T4		T5		V	A	B < D	B > D	
		<i>mo</i>	\tilde{x}	<i>mo</i>	\tilde{x}	<i>mo</i>	\tilde{x}							<i>mo</i>	\tilde{x}	<i>mo</i>	\tilde{x}	<i>mo</i>	\tilde{x}					
<i>growth rate</i>	15	4	4	4	4	5	5	14	10	20.0	<u>60.0</u>		<i>balancing act</i>	19	5	3	3	3	1	1.5	14	7	71.4	14.3
<i>flag carrier</i>	15	1	2	1	2	4	3	10	1	0.0	100.0		<i>future power</i>	19	5	3	1	3	4	3	13	6	16.7	50.0
<i>peace accord</i>	15	4	4	3	3	5	5	14	10	10.0	<u>70.0</u>		<i>love song</i>	19	5	5	5	4	5	5	16	15	6.7	40.0
<i>packaging product</i>	15	5	5	3	3	4	3.5	13	9	33.3	<u>44.4</u>		<i>car factory</i>	22	5	5	2	2.5	5	5	20	16	6.3	<u>75.0</u>
<i>power difficulty</i>	15	1	2	1	1.5	3	2.5	11	3	100.0	0.0		<i>health risk</i>	22	4	4	2	2	5	4	20	18	11.1	<u>77.8</u>
<i>risk disclosure</i>	15	1	1	1	1	3	3	10	1	100.0	0.0		<i>rumor control</i>	22	1	3	3	3	3	3	21	3	100.0	0.0
<i>security house</i>	15	3	3	3	3	5	4	11	6	33.3	<u>50.0</u>		<i>market depression</i>	22	1	2.5	2	2	1	2	20	5	60.0	20.0
<i>cigarette market</i>	19	5	4	3	3	4	3	17	9	33.3	<u>44.4</u>		<i>business information</i>	22	3	3	1	2	4	4	18	2	0.0	100.0
<i>failure risk</i>	19	5	3.5	2	2	1	3	15	6	50.0	16.7		<i>university center</i>	22	1	3	2	2	5	4	19	8	0.0	100.0
<i>plane crash</i>	19	5	5	5	4	5	5	13	12	25.0	<u>50.0</u>		<i>risk assessment</i>	22	1	1	3	3	3	3	18	2	50.0	50.0
<i>vehicle operation</i>	19	3	3	4	3	3	3	14	6	50.0	16.7		<i>sugar harvest</i>	22	1	1	1	2.5	4	4	17	3	33.3	33.3

results than the drawing. Despite this, it is interesting to see how some of the drawings are very similar to the blends (e.g. 5 of *love song*). On the other hand, the drawings 1 and 2 for *balancing act* were considered better than the blend, which shows that the system is not always capable of producing better solutions.

Discussion

The usefulness of a system can be assessed from several perspectives. In the introduction, we already addressed it using a research-interest point of view, showing the potential of the system in three different fields.

As already mentioned, our goal is not the generation of Unicode emoji. Despite this, we have shown the usefulness of the system in terms of complementing the emoji set (Cunha, Martins, and Machado 2018a), which still lacks in the coverage of several core concepts. For a list of 1509 core concepts, the system is able to produce concept-representative solutions for 1144 concepts, which is an improvement of 44.63% when compared to the results obtained by emoji set alone.

Our main goal is to produce visual representations of concepts. One question that arises is whether visual blending is suitable for concept representation (Q1). The high coverage of the core concept list is an argument in favour. However, when analysing the usage of the system by participants there are issues that point otherwise. First, the occurrence of emoji hiding – one of the emoji was partially or even totally hidden – which is an exploit of the system and does not make usage of visual blending. In study #1 hiding was observed in 7 out of the 39 exported blends. Another unfavourable result is the high occurrence of juxtaposition (Table 1), which we consider as a weak type of blending, having little advantage over a sequential positioning approach.

On the other hand, replacement as the most used type of blend is a good indication that the visual blending is beneficial. When analysing user drawings (obtained in a Pictionary-like task), users tend to draw existing objects and use a juxtaposition-based approach. Replacement often leads to metaphorical solutions, which normally require

more complex reasoning from humans. As such, the system provides a quick way to present the user with solutions that require such reasoning. These topics are related to Q2 and provide an indication that different blend types have different advantages.

The two biggest advantages of the system (Q3) are that it provides the user with the possibility of choosing among different blend type solutions, often leading to more than one solution deemed good (study #1), and that it follows a multi-purpose approach, allowing the user to introduce any concept without requiring changes to the configuration or extra input data.

As far as co-creativity is concerned, one of the most used arguments in favour of such systems is the capability of fostering users creativity (Liapis et al. 2016; Cunha, Martins, and Machado 2018a; Karimi et al. 2018). The interaction with the system allows the user to evolve solutions that match his/her preferences and, at the same time, both the user and the system are constantly influencing the perception of one another, leading to novel ideas. The results obtained in study #2 provide evidence that this interaction leads to better solutions than the ones drawn by the user alone. The potential of the system is even clearer if we consider that in two cases participants who knew the concept were not able to draw it and afterwards considered the blend as a good representation. Despite these results, there is still work to be done in assessing the impact of the co-creative functionalities on the user (e.g. suggestions made by the artificial evaluator) and also in making them more adequate to the needs of the user.

Overall, we believe that we have demonstrated the usefulness of the system in terms of (i) research purposes, (ii) emoji set completeness, (iii) visual representation of concepts, and (iv) creativity aiding and fostering.

Acknowledgements

This research is partially funded by Fundação para a Ciência e Tecnologia (FCT), Portugal, under the grants SFRH/BD/120905/2016 and SFRH/BD/132728/2017. This work includes data from ConceptNet 5, which was com-

plied by the Commonsense Computing Initiative and is freely available under the Creative Commons Attribution-ShareAlike license (CC BY SA 4.0) from conceptnet.io.

References

- Brysbaert, M.; Warriner, A. B.; and Kuperman, V. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior research methods* 46(3).
- Confalonieri, R.; Corneli, J.; Pease, A.; Plaza, E.; and Schorlemmer, M. 2015. Using argumentation to evaluate concept blends in combinatorial creativity. In *Proceedings of the Sixth International Conference on Computational Creativity*, 174–181.
- Cunha, J. M.; Gonçalves, J.; Martins, P.; Machado, P.; and Cardoso, A. 2017. A pig, an angel and a cactus walk into a blender: A descriptive approach to visual blending. In *Proceedings of the Eighth International Conference on Computational Creativity*.
- Cunha, J. M.; Lourenço, N.; Correia, J.; Martins, P.; and Machado, P. 2019. Emojinating: Evolving emoji blends. In *Proceedings of the Eighth International Conference on Computational Intelligence in Music, Sound, Art and Design*, 110–126. Springer.
- Cunha, J. M.; Martins, P.; and Machado, P. 2018a. Emojinating: Representing concepts using emoji. In *Workshop Proceedings of the Twenty-Sixth International Conference on Case-Based Reasoning*.
- Cunha, J. M.; Martins, P.; and Machado, P. 2018b. How shell and horn make a unicorn: Experimenting with visual blending in emoji. In *Proceedings of the Ninth International Conference on Computational Creativity, Salamanca, Spain, June 25-29, 2018*.
- Davis, N.; Hsiao, C.-P.; Singh, K. Y.; and Magerko, B. 2016. Co-creative drawing agent with object recognition. In *Twelfth artificial intelligence and interactive digital entertainment conference*.
- Fares, M. 2016. A Dataset for Joint Noun-Noun Compound Bracketing and Interpretation. In *Proceedings of the ACL 2016 Student Research Workshop*, 72–79. Association for Computational Linguistics.
- Ha, D., and Eck, D. 2017. A neural representation of sketch drawings. *arXiv preprint arXiv:1704.03477*.
- Karimi, P.; Maher, M. L.; Grace, K.; and Davis, N. 2018. A computational model for visual conceptual blends. *IBM Journal of Research and Development*.
- Liapis, A.; Hoover, A. K.; Yannakakis, G. N.; Alexopoulos, C.; and Dimaraki, E. V. 2015. Motivating visual interpretations in iconoscope: Designing a game for fostering creativity. In *Proceedings of the Foundations of Digital Games Conference*.
- Liapis, A.; Yannakakis, G. N.; Alexopoulos, C.; and Lopes, P. 2016. Can computers foster human users' creativity? theory and praxis of mixed-initiative co-creativity. *Digital Culture & Education* 8(2):136–153.
- Phillips, B. J., and McQuarrie, E. F. 2004. Beyond visual metaphor: A new typology of visual rhetoric in advertising. *Marketing theory* 4(1-2):113–136.
- Puyat, M. 2017. Emotigan: Emoji art using generative adversarial networks. CS229: Machine Learning Course, Stanford University.
- Radpour, D., and Bheda, V. 2017. Conditional generative adversarial networks for emoji synthesis with word embedding manipulation. *arXiv preprint arXiv:1712.04421*.
- Speer, R., and Havasi, C. 2012. Representing general relational knowledge in conceptnet 5. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, 3679–3686.
- Wijeratne, S.; Balasuriya, L.; Sheth, A.; and Doran, D. 2017. Emojinet: An open service and api for emoji sense discovery. In *Proceedings of ICWSM-17*.
- Xiao, P., and Linkola, S. 2015. Vismantic: Meaning-making with images. In *Proceedings of the Sixth International Conference on Computational Creativity, ICC-15*.
- Yannakakis, G. N.; Liapis, A.; and Alexopoulos, C. 2014. Mixed-initiative co-creativity. In *Proceedings of the Ninth Conference on the Foundations of Digital Games*.

A Dynamic Approach for the Generation of Perceptual Associations

Ana Rodrigues, Amilcar Cardoso, Penousal Machado

CISUC, Department of Informatics Engineering
University of Coimbra
Coimbra, Portugal
{anatr, amilcar, machado}@dei.uc.pt

Abstract

We propose a dynamic approach and computational artefact for the generation of cross-modal associations between the domains of visual communication and perception of emotions. This is accomplished by applying three key steps: (i) assemble an alphabet (ii), define a shape grammar, and (iii) generate computationally a dynamic representation for the previous point. We restrain the visual expression to a line movement and limit our set of emotions to four in order to establish an initial solid baseline of experiments. Experiments were done to assess the validity of the established associations between emotions and visual features. Results have confirmed some associations previously suggested by literature and have emphasized others. This approach may be a key aspect to solve problems of multi-domain contexts where non-verbal communication is required to connect two or more domains, e.g. visual arts, artistic performances, non-verbal languages, and entertainment.

Introduction

Emotions are shared among humans and transversal to several contexts and situations of our daily lives. They have been quite a popular means to mediate findings from distinct domains of knowledge (Lindborg and Friberg 2015; Pesek et al. 2017). Based on theoretical and practical evidence, we suggest that crossing perceptual and cognitive findings with several aspects of images (shape, size, etc.) may bring an additional value to a non-verbal communication and computational generation of multi-modal associations. To assess this, we develop a solution based on three-step key concepts: gather a multidisciplinary alphabet, define a shape grammar, and develop a computational solution.

The use of computational tools to explore multi-modal associations has been a topic of growing interest. This can be justified by the computers capacity to interpret and evolve abstract concepts with a certain degree of autonomy and abstraction (Boden and Edmonds 2009; Noll 1966; Moroni and others 2014; Sims 1992). Still, it remains a challenge in combining empirical knowledge and computational setups to create configurations with a certain degree of creativity and plasticity (McCormack et al. 2009).

Regarding the set of emotions used for this project, we decided to restrain our experiment setup to four emotions:

anger, calm, happy, and sad the four extremes of the two-dimensional model of emotions (Russell 1980).

We defined as well set of rules to unambiguously guide a visual representation of these - in this stage of work we restrained this to visual expression of a line movement as it is a strong element of expression in any visual composition. We founded our computational language in shape grammars' knowledge. A shape grammar is a set of rules and geometric transformations that represent pre-defined rules. As (Stiny 1980) describes, a shape grammar (SG) is a 4-tuple " $sg = (vt, vn, r, i)$ " in which (i) vt : a set of terminal symbols; (ii) vn : a set of non terminal symbols; (iii) r : a finite set of ordered pairs whereas a shape consisting of an element of vt is combined with an element of vn ; (iv) i : an initial shape structure consisting of elements of vt and vn . Shape grammars not only provide a great plasticity but also allow to keep a visual "identity (aesthetic coesion) in the representation of an abstract morphology (Stiny 1975).

Regarding the computational generation of these rules, we did look at the problem from a physics point of view. We applied a physics approach to modulate the line motion and therefore obtain a visual expression resulting from this. For this reason, we restrain the graphic representation of associations to the vectorial domain. As a clarification, our main interest is not to test an isolated visual feature but instead a combination of features or visual aspects.

Our main contribution consists in providing a solution to a non-verbal visual communication for multi-domain contexts, which in turn can be used to motivate/inspire creative applications among others.

Related Work

Abstract forms of non-verbal communication have been widely explored by expressionist painters like Wassily Kandinsky (Kandinsky and Rebay 1979) and Paul Klee (Klee and Moholy-Nagy 1953). Along with their visual interpretations, they shared the belief that certain combinations of color, light and form would enhance the visual experience.

Aiming to find an empirical way of bridging artistic domains (like music or visual expression) and human perception several models of emotion have been proposed (Russell 1980; Hevner 1936; Brattico and Pearce 2013). The most popular models are the dimensional and discrete

models of emotions. Dimensional models have attempted to identify a series of dimensions capable of representing all possible emotional states (Brattico and Pearce 2013; Buechel and Hahn 2017). To overcome the limitations of dimensional models, several authors have proposed discrete models of emotion that use other categories to classify emotion and some additional categories as well (Brattico and Pearce 2013; Buechel and Hahn 2017; Scherer 2000).

Most studies found in literature relating emotions and visual aspects have limited their experiments to the visual features of color (hue, brightness, saturation) and shape. In general, brighter colors have been associated with positive emotions whereas darker colors have been associated to negative emotions (Marks 1996; Barbiere, Vidal, and Zellner 2007). Regarding form, round shapes have been mostly associated to positive emotions like calm and pleasure whereas sharp shapes have been linked to negative emotions like fear and anger (Cavanaugh, MacInnis, and Weiss 2016; Ramachandran and Hubbard 2001). Particularly, Cavanaugh, MacInnis, and Weiss have conducted an interesting study on the use of multiple perceptual dimensions to differentiate emotions.

On the computational application of these cross-modal associations, examples of perceptually based tools relating auditory and visual dimensions can be found in the works of Grill and Flexer and Lindborg and Friberg. While Grill and Flexer (2012) relied on an exploratory visualization to produce sound with perceptually relevant textural metaphors, Lindborg and Friberg (2015) investigated associations that may arise at emotional levels on modal correspondences between certain sounds and colour.

Vocabulary and Grammar Rules

In this section we present a set of visual features and rules to manipulate and transform their expression. The definition of the vocabulary and grammar rules are key aspects, as we propose a grammar based approach for the computational generation of the image-emotion associations.

The Dataset

A Literature research on emotions and visual features was conducted (Scheerer and Lyons 1957; Karwoski, Odbert, and Osgood 1942; Cavanaugh, MacInnis, and Weiss 2016; Lyman 1979; Poffenberger and Barrows 1924; Collier 1996; Lindborg and Friberg 2015; Peters and Merrifield 1958) to collect a dataset of relevant features and therefore build an alphabet of abstract visual symbols.

Inspired by the visual grammar proposed by Leborg, we gathered a set of visual features that can be organized into distinct sections. These include features such as shape, color, size, texture, stroke, contours, density, visual complexity, visual distribution, balance, and geometric transformations. However, as we previously said we restrained this study to a vectorial expression of line and so we do not explore features such as texture, other shapes, closure, high-level features, the geometric transformations of scale and reflect.

Graphic Representations

The movement of line was determined by (i) time (duration and repetition of events) and (ii) a set of visual features. Time determined whether the current visual configuration was repeated or not. A visual demonstration of these can be found in figure 1.

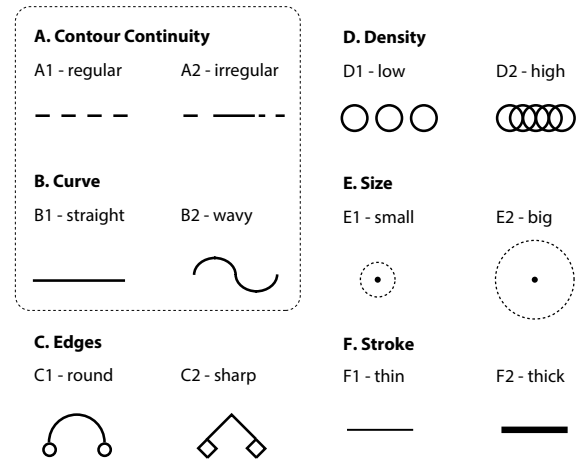


Figure 1: Visual morphology.

A “regular” classification means that the current visual motif repeats itself in regular periods of time and “irregular” means that the current visual motif will change over irregular periods of time. The visual features approach the following aspects: continuity, curve, edge, density, size, and stroke. The reason for introducing the aspect of time and motion in image is that in the future work we pretend to integrate musical aspects as well.

“Straight” classification was interpreted as a constant, i.e., no variations in angular velocity except in the case of irregular continuity. “Wavy” was interpreted as line movement with variations in angular velocity. They could range from soft changes - if we want to achieve a “round” path - to abrupt changes - if we want to achieve a “sharp” path.

“Density” was interpreted as the distance between the current location of the line and the next location, i.e., a smaller distance will produce a higher density and vice versa.

“Size” was visually interpreted as the diameter surrounding the current location of line when compared to the baseline. In future experiments size may be interpreted as the length of the line as well.

“Stroke” was represented as the thickness of line.

A Dynamic Computational Interpretation

We propose a visual interpretation inspired by the laws of physics on motion (linear velocity, acceleration, and angular velocity). For the purpose of this study linear velocity and acceleration can be held a constant or vary over time. The physical attribute that is most relevant in this case is angular velocity. Table 1 gives an overview on the type of changes that are made in order to produce the desired visual motion.

Table 1: Translation of physical dimensions to a *line motion*.

<i>Morphology</i>	<i>Lin Vel & Acc</i>	<i>Angular Velocity</i>
<i>contour</i>	<i>regular</i>	constant
	<i>irregular</i>	irregular change
<i>curve</i>	<i>straight</i>	constant
	<i>wavy</i>	constant or variable
<i>edges</i>	<i>round</i>	soft or abrupt change
	<i>sharp</i>	soft change abrupt change

The Generative Effect Although we follow a set of grammar rules our system has a generative side. This happens because the variables of each visual parameter (size, density, curve, edge, and stroke) are chosen from a random of values within a specific range. For example, for big size, the size output may be a float that can range from “3” to “6”. That is, we defined minimum and maximum values for each category (e.g. small, medium, big) of a visual feature.

The Experiment

To validate the previously established associations and combinations of visual features for the line motion, we conceived an experiment. In this section we explain the methodology used in the conception of our experiment.

User Study

The user study consisted of two parts. The first, where an image was presented - that we believed to be the most representative of a certain emotion - we asked participants to identify the emotion presented in it. The second, consisted of observing four possible visual expressions animated to represent a specific emotion and choosing the most representative image regarding an emotion at a time. In both cases, participants had the possibility to choose the option of “none of the above” in case they considered that none of the options was good enough.

Visual Setup

The choice for the images to be presented was based on literature findings but also on personal choices. The reasons for the latter, of adding features characteristics that were not presented in literature, was that we wanted to test as many possibilities as we could and had to adjust others to our visual interpretation data in order to promote expressive outputs. The visual configurations used in the first part of the experiment can be consulted as well in table 2.

Visual features that weren’t mentioned in literature but were also considered were inspired by non empirical works of Visual Artists like Oskar Fischinger and Abstract Painters.

Results

This experiment gathered the opinion of a total of 45 participants. The average age was 31.8 with a standard deviation of 10.3. Out of the 45 participants, 51.1% were female and 48.9% were male. Additionally, we questioned whether they had a background in visual communication or not, to which

Table 2: Experiment setup: Part 1.

<i>E</i>	<i>Cont.</i>	<i>Curve</i>	<i>Edge</i>	<i>Den.</i>	<i>Size</i>	<i>Stroke</i>
<i>anger</i>	irreg.	wavy	sharp	high	big	med.
<i>calm</i>	reg.	wavy	round	low	med.	thick
<i>happy</i>	irreg.	wavy	round	med.	big	thin
<i>sad</i>	irreg.	wavy	round	low	small	thick

33.3% answered “yes” and 66.7% answered “no”. In this section we presented the results obtained both in Part 1 and 2 of the experiment.

Part 1

In “Part 1” of the study we aimed to evaluate whether the participant’s choice of emotion did correspond to a specific animated expression of the line (through a pre-defined setup of visual configurations (see table 2) or not.

The visual configurations that succeeded the most, regarded the visual representation of the following emotions: “calm” (77.8%), “anger” (71.1%), and “happy” (68.9%) respectively. Surprisingly, the same combination of features in the representation of “anger” in the second part of the test was chosen by a slightly smaller number of participants (46.7%). “Sad” (35.6%) was the emotion where participants had the biggest difficulty to associate with the presented animation. This is congruent with results obtained in the second part of the experiment where this same representation of sadness was chosen by 26.7% of participants (see table 3 and table 4).

Additionally, associations were made to other emotions than the expected. For instance, “Anger” and “Calm” visual representations were associated with “happy” and “sad” emotions. “Happy” visual representation was associated to “calm” and “sad” emotions. “Sad” was associated to “calm” and “anger” emotions. In spite of this, we should note that these associations to other emotions than the desired ones were made by a small amount of participants (no more than 17.8%, see table 3).

Overall, all the other emotion classification by participants matched the desired representations, except in the case of sadness (see table 3).

Table 3: Experiment results: Part 1.

<i>Emotion</i>	<i>Expected</i>	<i>None</i>	<i>Other</i>
<i>anger</i>	71.1%	22.2%	6.7%
<i>calm</i>	77.8%	4.4%	17.8%
<i>happy</i>	68.9%	13.3%	17.8%
<i>sad</i>	35.6%	57.8%	6.6%

Part 2

Looking at table 4 at the results of part 2 of the experiment we can see the most preferred visual configurations for emotions by participants.

“Anger” stands out from the remaining emotions because it had two expressions with divided opinions among participants (46.7% and 40%). The difference between them lies on the features of density, size and in this particular case of visual interpretation of “wavy” and “sharp” line. While on the first case a circular motion with sharp edges was applied (see in table 4 anger a), on the second case (see in table 4 anger b) it was generated a wavy motion with sudden changes of direction and therefore sharp edges.

“Calm” and “Happy” chosen visual configurations (62.2% and 75.6% respectively) were congruent with the visual setup of part 1, meaning that it has been found a satisfactory combination of visual features.

As for the “Sad” emotion visual configuration, even though it was chosen approximately by half the population (48.9%) which isn’t a choice by great majority. When we compare this to the results of others and consider the results of Part 1 of the study as well, it leads us to think that we found a better visual representation for this emotion.

Table 4: Experiment results: Part 2.

E	p%	Cont.	Curve	Edge	Den.	Size	Strk.
<i>anger</i>	a:46.7	irreg.	wavy	sharp	high	big	med.
	b:40	irreg.	wavy	sharp	med.	med.	med.
<i>calm</i>	62.2	reg.	wavy	round	low	med.	thick
<i>happy</i>	75.6	irreg.	wavy	round	med.	big	med.
<i>sad</i>	48.9	reg.	straight	—	—	—	thick

General Findings

Overall, the same combinations for visual features to represent emotions oscillated a bit between part 1 and part 2 of the study. When compared to first part, in the second part there was a tendency to decrease the choice over the same visual configurations presented in part 1. The only exception to this was with the emotion “Happy”, because participants enhanced this option by increasing their choice in 6.7% comparatively to part 1. Despite these oscillations, calm seemed to be the emotion with the visual representation more accepted among all the participants. As for the visual representations of “anger” and “sad” (see 2) users preferred with a slightly different visual configurations. The visual representation of “sad” was the one that least matched the previously established findings.

By comparing the results obtained in experiments with previously established associated in literature (Scheerer and Lyons 1957; Karwoski, Odbert, and Osgood 1942; Cavanaugh, MacInnis, and Weiss 2016; Lyman 1979; Poffenberger and Barrows 1924; Collier 1996; Lindborg and Friberg 2015; Peters and Merrifield 1958), we confirm the following findings between visual features and emotions: (i) “anger” associates to wavy, sharp, high density, big size and medium stroke; (ii) “calm” associates to regular contour, round edge, thick stroke; (iii) “happy” associates to round edge, big size, and thin stroke; (iv) “sad” associates to regular contour and thick stroke. Visual features that weren’t mentioned in literature but were also considered relevant in

the experiments done in our study concerned the following features: irregular continuity in the case of “anger”; wavy curve, low density and medium size in the case of “calm”; irregular contour, wavy curve, and medium density in the case of “happy”; straight curve and low density in the case of “sad”.

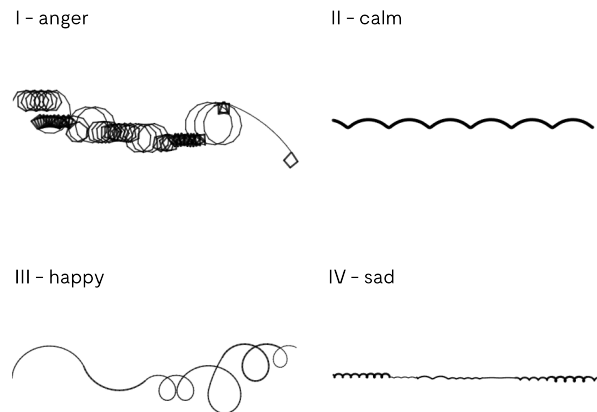


Figure 2: Visual configurations presented in the first part of the experiment. As these images are animated gifs, we made them available in the following url: <https://bit.ly/2GQoB8p>

Further Explorations

In future work we aim to explore more visual features and apply an Evolutionary User Guided Algorithm to evolve and explore new combinations of visual features. This evolutionary approach has two benefits: the generation of associations that suit the preferences of the user (% of being chosen that a visual feature has regarding a specific emotion); and the analysis of the interactions of the use that may allow a better understanding on the users perceptual motivations.

Other aspects of concern relate to the measurement of some visual elements that are not trivial, as it is the case of the complexity of an image for instance. Another aspect that may be relevant in the future is related to the degree of intensity of an emotion (e.g., joy vs happy or sad vs devastated), i.e., how different degrees of intensity are represented visually.

Conclusion

We have presented a system that is capable of generating dynamically associations between a set of visual features (translated into a line expression) and emotions. We did user tests to validate and tested these multi-modal associations. Results of the system output were satisfactory in the sense that they confirmed the perceptually established associations between line features and emotions. Moreover, we suggested new associations between emotions and visual features that weren’t suggested on experiments from other authors. Afterwards, we discussed some challenges faced in the development and exploration of such an interdisciplinary work as well as improvements to be done in future work such as the introduction of an evolutionary user guided algorithm.

Acknowledgments

This work was supported by the Portuguese National Foundation for Science and Technology under the grant SFRH/BD/139775/2018.

References

- Barbieri, J. M.; Vidal, A.; and Zellner, D. A. 2007. The color of music: Correspondence through emotion. *Empirical studies of the arts* 25(2):193–208.
- Boden, M. A., and Edmonds, E. A. 2009. What is generative art? *Digital Creativity* 20(1-2):21–46.
- Brattico, E., and Pearce, M. 2013. The neuroaesthetics of music. *Psychology of Aesthetics, Creativity, and the Arts* 7(1):48.
- Buechel, S., and Hahn, U. 2017. A flexible mapping scheme for discrete and dimensional emotion representations: Evidence from textual stimuli. In *CogSci 2017 Proceedings of the 39th Annual Meeting of the Cognitive Science Society*, 180–185.
- Cavanaugh, L. A.; MacInnis, D. J.; and Weiss, A. M. 2016. Perceptual dimensions differentiate emotions. *Cognition and Emotion* 30(8):1430–1445.
- Collier, G. L. 1996. Affective synesthesia: Extracting emotion space from simple perceptual stimuli. *Motivation and emotion* 20(1):1–32.
- Grill, T., and Flexer, A. 2012. Visualization of perceptual qualities in textural sounds. In *ICMC*. Citeseer.
- Hevner, K. 1936. Experimental studies of the elements of expression in music. *The American Journal of Psychology* 48(2):246–268.
- Kandinsky, W., and Rebay, H. 1979. *Point and line to plane*. Courier Corporation.
- Karwoski, T. F.; Odbert, H. S.; and Osgood, C. E. 1942. Studies in synesthetic thinking: Ii. the role of form in visual responses to music. *The Journal of General Psychology* 26(2):199–222.
- Klee, P., and Moholy-Nagy, S. 1953. *Pedagogical sketchbook*. Faber & Faber London.
- Leborg, C. 2006. *Visual grammar*. Princeton Architectural Press.
- Lindborg, P., and Friberg, A. K. 2015. Colour association with music is mediated by emotion: evidence from an experiment using a cie lab interface and interviews. *PLoS one* 10(12):e0144013.
- Lyman, B. 1979. Representation of complex emotional and abstract meanings by simple forms. *Perceptual and Motor Skills* 49(3):839–842.
- Marks, L. E. 1996. On perceptual metaphors. *Metaphor and Symbol* 11(1):39–66.
- McCormack, J.; Eldridge, A.; Dorin, A.; and McIlwain, P. 2009. *Generative algorithms for making music: emergence, evolution, and ecosystems*. na.
- Moroni, A., et al. 2014. Transgenic visual-and-sound compositions. *NICS Reports* (9):8.
- Noll, A. M. 1966. Computers and the visual arts. *Design Quarterly* 64–71.
- Pesek, M.; Strle, G.; Kavčič, A.; and Marolt, M. 2017. The moodo dataset: Integrating user context with emotional and color perception of music for affective music information retrieval. *Journal of New Music Research* 46(3):246–260.
- Peters, G. A., and Merrifield, P. R. 1958. Graphic representation of emotional feelings. *Journal of clinical psychology* 14(4):375–378.
- Poffenberger, A. T., and Barrows, B. 1924. The feeling value of lines. *Journal of Applied Psychology* 8(2):187.
- Ramachandran, V. S., and Hubbard, E. M. 2001. Synaesthesia—a window into perception, thought and language. *Journal of consciousness studies* 8(12):3–34.
- Russell, J. A. 1980. A circumplex model of affect. *Journal of personality and social psychology* 39(6):1161.
- Scheerer, M., and Lyons, J. 1957. Line drawings and matching responses to words 1. *Journal of Personality* 25(3):251–273.
- Scherer, K. R. 2000. Psychological models of emotion. *The neuropsychology of emotion* 137(3):137–162.
- Sims, K. 1992. Interactive evolution of dynamical systems. In *Toward a practice of autonomous systems: Proceedings of the first European conference on artificial life*, 171–178.
- Stiny, G. N. 1975. Pictorial and formal aspects of shape and shape grammars and aesthetic systems.
- Stiny, G. 1980. Introduction to shape and shape grammars. *Environment and planning B: planning and design* 7(3):343–351.

Performing Structured Improvisations with Pre-trained Deep Learning Models

Pablo Samuel Castro

Google Brain
psc@google.com

Abstract

The quality of outputs produced by deep generative models for music have seen a dramatic improvement in the last few years. However, most deep learning models perform in “of-line” mode, with few restrictions on the processing time. Integrating these types of models into a live structured performance poses a challenge because of the necessity to respect the beat and harmony. Further, these deep models tend to be agnostic to the style of a performer, which often renders them impractical for live performance. In this paper we propose a system which enables the integration of out-of-the-box generative models by leveraging the musician’s creativity and expertise.

Introduction

The popularity and quality of machine learning models has seen a tremendous growth over the last few years. *Generative models*, which are trained to produce outputs resembling a pre-specified data distribution, have attracted much attention from both the scientific and artistic community in large part due to the realism of the outputs produced.

In the musical domain, recent works produce music that is both realistic and interpolatable (Roberts et al., 2018), closely resembles human performance (Huang et al., 2019), and can aid in automatic composition¹. The increased realism of these models is typically accompanied with an increase in the amount of processing time required to generate outputs. Unfortunately, long processing times generally renders these models inadequate for live performance. This issue is particularly stark in structured improvisation, such as in traditional jazz, where the music produced must respect the beat and harmony of the piece.

In this paper we introduce a software system that enables the incorporation of generative musical models into musical improvisation. This can be used as both a solo-performance or in an ensemble. Our system produces a performance that is a hybrid of human improvisation with melodies and rhythms generated by deep learning models. Our hybrid approach enables us to address real-time compatibility and stylistic personalization.

¹<https://www.ampermusic.com/>

Background

We use Recurrent Neural Networks (RNNs) (Rumelhart, Hinton, and Williams, 1986) as the machine learning models for generating drum beats and melodies. Recurrent Neural Networks are a special type of neural network which process a *sequence* of tokenized inputs one token at a time, updating an internal state after processing each input. A trained RNN can be used for generation: after processing a sequence of tokens $t_{1:n}$, sample from the resulting internal distribution over the token dictionary. When using these models for generation, we will refer to the initial sequence $t_{1:n}$ fed into the model as the *primer sequence*.

We will make use of two LSTM-models from Google Magenta. The first is MelodyRNN (Magenta, 2016b). It processes *note events* as tokens, where a note event contains a note’s pitch and its duration. The model assumes monophonic melodies (i.e. only one note played at a time) and is instrument agnostic. Thousands of MIDI files were used for training. These MIDI files were quantized into 16th notes: that is, the minimum allowable time between two notes are one 16th note². When using this model to generate new melodies, the melodies produced tend to match the key signature and note density of the primer melody sequence fed into it, which is a desirable property for our use case. The second is DrumsRNN (Magenta, 2016a). The model is similar to MelodyRNN, but here there is polyphony as multiple drums can be hit simultaneously. As for MelodyRNN, this model was trained on thousands of MIDI files, quantized into 16th notes.

Related Work

There have been a number of works proposing new types of digital instruments which make use of machine learning models. The Wekinator (Fiebrink, 2009) enables users to train new models in a *supervised* fashion by providing pairs of inputs and expected outputs; inputs can be provided in many forms including using computer controllers and physical gestures, while outputs can be sent to any musical, digital or physical actuator. This contrasts with our proposed framework, which does not require retraining a model, but rather adapt the outputs of a pre-trained deep learning model to a performer’s style.

²There are sixteen 16th notes in one bar of 4/4 time.

Thom (2000) and Thom (2001) build probabilistic models to emulate an improviser’s tonal and melodic trends. Johnson, Keller, and Weintraut (2017) makes use of two LSTMs: one for intervals between notes and the other for note intervals relative to the underlying chord progression; these trained models are then combined to generate melodies in a recurrent note-by-note fashion. In (Weinberg et al., 2009) the authors introduce *shimon*, a robot marimba player capable of interacting with human players. The robot has human-like movements (such as head-bobbing, “gazing” to pass on the solo to another player, etc.) which make it natural to interact with. Closely related to our use of ‘continuations’ are *The Continuator* of Pachet (2003), where the authors use Markov models to adapt to a user’s style. In contrast to our work, however, the continuer is agnostic to the underlying beat of a performance, which is essential to jazz improvisation. Bretan et al. (2017) propose training a deep autoencoder to encode melodies played by a performer into a latent space that has been trained to capture musical consistency; the closest melody from a library that has been embedded into the same latent space is returned, allowing their system to respond in near real-time. Roberts et al. (2018) propose a deep autoencoder model for encoding melodies into a latent space, combined with a deep decoder for converting points from that latent space into cohesive melodies. Huang et al. (2019) trained a transformer model (Vaswani et al., 2017) on a dataset of virtuoso piano performances, resulting in a model that can produce highly realistic and novel musical snippets.

System setup

Our setup assumes a piano keyboard connected to a computer via MIDI used for input, along with an additional controller for enabling more MIDI control messages; in our case we are using the Korg Nanokontrol2 MIDI controller but the system can be used with any MIDI controller. We use SuperCollider³ to detect all incoming MIDI events and pipe them as OSC⁴ messages to a Python backend running on the same machine. The Python backend processes the notes and may then send an OSC message containing notes to be played to SuperCollider, which either generates the sound or forwards them to an external MIDI controller for producing the sound.

The SuperCollider component acts mostly as a bridge between the MIDI controllers and the Python backend. It defines a set of handlers for routing MIDI input messages to the backend via OSC messages, and handles OSC messages from the backend. When a note on/off message is received from the backend, it can either redirect to an external MIDI controller or produce the sound itself. For the latter, the SuperCollider code loads a set of WAV files as well as a few synthetic instruments for playback.

Backend design

At its core, the Python backend is running a continuous loop over a customizable number of bars, each with a customizable number of beats. Each is discretized it into 16th note

³<https://supercollider.github.io/>

⁴<http://opensoundcontrol.org/>

segments (so one bar in 4/4 time signature will have 16 intervals). Multi-threading is used to allow for real-time response, and we maintain a set of global variables that are shared across the different threads, the most important of which are listed below:

- **time_signature**: An object containing a pair of integers denoting the *numerator* (4, 6, 7, etc.) and *denominator* (4, 8, or 16) of the time signature.
- **qpm**: A float indicating the speed (quarters-per-minute) of playback. One quarter note is equal to four 16th notes, so this value indicates the time needed to process four 16th note events.
- **playable_notes**: A SortedList where we store each playable note event. Each element contains the type of playback event (click track, bass, drums, etc.), the note pitch, the instrument itself (bass, keyboard, hi-hat, bass drum, crash, etc.), and the 16th note in the bar where the event occurs.
- **bass_line**: Similar to *playable_notes* but containing only the current bassline.
- **accumulated_primer_melody**: A list which will accumulate the note pitches played by the human improviser. Once enough notes have been accumulated they will be sent as a ‘primer’ melody to MelodyRNN. This is discussed in more detail in the Improvisation section.
- **generated_melody**: A list containing the note pitches produced by MelodyRNN. When full, the note pitches played by the human will be replaced by the pitches in this buffer.

The open source-code can be accessed at <https://github.com/psc-g/Psc2>.

Click-track generation

The first step is setting the number of bars, time signature, and tempo (qpm). The user may change the number of bars, time signature numerator, and time signature denominator via a set of buttons on the Nanokontrol2. The qpm may be adjusted via a knob or by tapping the beat on a button. These define the length and structure of the *sequence*, which the system will loop over. Once these are set the user may start playback by hitting the ‘play’ button on the Nanokontrol2. This will start a click-track which will make use of 3 different click sounds:

1. The first will play on the first beat of the first bar, to indicate the start of the sequence. This is important for the user to know the start of the sequence when recording a bassline or chords.
2. The second will play on the first beat of the remaining bars in the sequence (if at least two bars were selected)
3. The third will play within each bar at a frequency marked by the time signature denominator: if the denominator is 4, it will play a click every four 16th notes; if it is 8, it will play every two 16th notes; if it is 16 it will play a click every 16th note.

Once the click-track has been started, the user can place the system in one of four *modes* via buttons on the Nanokontrol2. When SuperCollider is in charge of producing sounds, each mode uses a different instrument for playback.

- **bass:** The user can record a bassline which will be looped over. After entering this mode, recording begins as soon as a note is pressed and proceeds until the end of the sequence is reached.
- **chords:** The user can play a set of chords to include in the loop playback. As in bass mode, recording begins as soon as a note is pressed and proceeds until the end of the sequence is reached.
- **improv:** Used for improvising over the loop playback in a call-and-response between the human and the machine learning model. This mechanism is discussed in more detail in the Improvisation section.
- **free:** Free-play mode, where the human can improvise freely over the loop playback.

Drums generation

Our system generates two types of drum beats: a deterministic one and another which is generated by a machine learning model. The deterministic one is built off of the bassline as follows:

1. A bass drum note is added at the first beat of every bar.
2. A snare note is added at each bass note onset.
3. Hi-hat notes are added at each 8th note (so every two 16th notes).

By pressing one of the Nanokontrol2 buttons, this deterministic drum beat is fed into DrumsRNN as a ‘primer’ to produce a new beat. Figure 1 illustrates this process in musical notation.

Improvisation

The improvisational part of our system is inspired on the call-and-response improvisations that are common in traditional jazz. In these sections two or more musicians take turns improvising over the same piece, and each musician usually incorporates melodies and/or rhythms played by previous musicians into their improvisations.

There are two main components to an improvisation: the pitches chosen and the rhythm of the notes. In our experience playing with generative models, such as MelodyRNN, we found that the rhythm of the melodies produced is not very reflective of the types of rhythms observed from professional improvisers. This may be due in large part to the 16th note quantization that is necessary for training the models. To overcome this issue, we propose a hybrid approach: the machine learning models provide the pitches, while the human provides the rhythm.

The way this is achieved is as follows:

1. Collect the pitches played by the human improviser in the *accumulated_primer_melody* global buffer.
2. Once the number of notes in the buffer is above a pre-specified threshold, the buffer is fed into MelodyRNN as a primer melody in a separate thread.

Figure 1: Building the drum beats. From top-to bottom: starting from a specified bassline, bass drum notes are added on the first beat of each bar, snare drum notes are added for each bass-note onset, and finally hi-hat notes are added at each 8th note. This deterministic drum beat can then be sent as a ‘primer’ to DrumsRNN which will generate a new beat.

3. When the MelodyRNN thread has finished generating a new melody, it will store *only the pitches* in the *generated_melody* buffer (the rhythmic information is dropped).
4. When the main looper thread detects that the *generated_melody* buffer has been filled, it will **intercept** incoming notes played by the user and replace their *pitches* with the pitches stored in *generated_melody* (and removing said pitch from the buffer). Figure 2 illustrates this process.
5. Once *generated_melody* is empty, return to step 1.

Our hybrid approach to machine-learning based improvisation allows us to mitigate the two problems mentioned in the introduction: real-time compatibility and stylistic personalization. The former is handled by performing the inference in a separate thread and only using it when it is available. The latter is handled by maintaining the rhythmic inputs from the human performer. It has been found that rhythm can significantly aid in facilitating melody detection (Jones, 1987), which we believe also carries over to enhancing the personalized style of performance. Further, by leveraging the human’s rhythmic input, we are able to avoid having the limitation of the 16th-note quantization that the RNN models require.

We provide some videos demonstrating the use of this system at <https://github.com/psc-g/Psc2/tree/master/research/nips2018>.

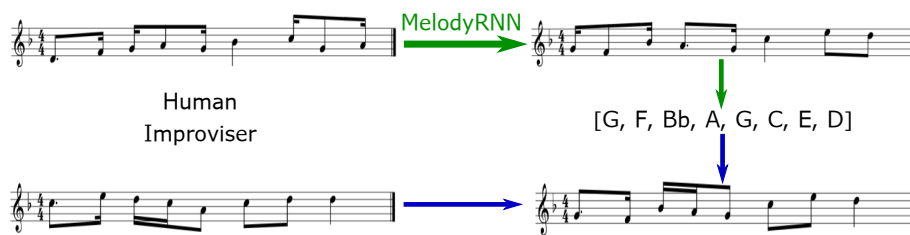


Figure 2: Building the hybrid improvisation. 1. The melody from the human improviser (top-left) is fed into MelodyRNN. 2. MelodyRNN produces a new melody (top-right). 3. The human improviser plays a new melody (bottom-left). 4. A new hybrid melody is created by combining the pitches from the MelodyRNN model with the rhythm from the most recent human improvisation (bottom-right).

Evaluation

Our proposed system has been used for live jazz performance in a piano-drums duet. The songs performed were some that the duo regularly plays, and our system was engaged during the improvisation sections of these songs. Since there was a human drummer performing, only the improvisation (MelodyRNN) part of our system was used. We report the feedback received from these performances.

Strengths

- The system was able to respond in real-time.
- The audience (many of which are familiarized with the pianist’s style) reported not noticing that there was an external system affecting the improvisations (they were only made aware of it after the show).
- The pianist felt creatively challenged when improvising with the system engaged, which led to a different way of playing.

Weaknesses

- The system did not work well on songs with many harmonic changes.
- The system would sometimes break up the pianist’s lines before they were done.
- The system would sometimes jump octaves when engaging.

Conclusion and Future Work

In this paper we have introduced a system that enables the integration of out-of-the-box deep learning models for live improvisation. We have designed it in a way that it does not require machine learning expertise to use, and can be extended to other types of musical generative models with little effort. Our hybrid approach for generating machine-learning based improvisations maintains the style of the human improviser while producing novel improvised melodies.

Although our system was built with MelodyRNN and DrumsRNN, the setup can be used with any musical generative model with relatively little effort. Along these lines, one avenue we would like to explore in the future is the incorporation of models which do not require quantization, such as PerformanceRNN (Simon and Oore, 2017); one challenge is

to ensure that the personal style of the human improviser is maintained.

Expert musicians are able to produce high-quality improvisations *consistently* from having honed their craft over many years of practice. A common frustration with these artists, however, is that they often find their improvisations too predictable, and struggle escaping their “pocket”. Our hope is that systems like the one we are proposing here can push expert musicians, and artists in general, out of their comfort zone and in new directions they may not have chosen to go to on their own. The experience of the pianist we reported in the previous section perfectly showcases this.

We hope to improve the system by allowing the performer to have more control over when the system begins recording, and when the system replaces the notes. We have already begun experimenting with this extra modality using a MIDI footpedal. Initial experiments suggest this added level of control mitigates for many of the issues raised by the human performer regarding the timing of when the system is engaged, while maintaining the novelty of the melodies produced.

References

- Bretan, M.; Oore, S.; Engel, J.; Eck, D.; and Heck, L. 2017. Deep Music: Towards Musical Dialogue. In *AAAI*, 5081–5082.
- Fiebrink, R. 2009. Wekinator. <http://www.wekinator.org/>.
- Huang, C. A.; Vaswani, A.; Uszkoreit, J.; Shazeer, N.; Hawthorne, C.; Dai, A. M.; Hoffman, M. D.; and Eck, D. 2019. An Improved Relative Self-Attention Mechanism for Transformer with Application to Music Generation.
- Johnson, D. D.; Keller, R. M.; and Weintraut, N. 2017. Learning to Create Jazz Melodies Using a Product of Experts. In *Proceedings of the The Eighth International Conference on Computational Creativity*.
- Jones, M. R. 1987. Dynamic pattern structure in music: Recent theory and research. *Perception & Psychophysics* 41(6):621–634.
- Magenta, G. 2016a. Drumsrnn. https://github.com/tensorflow/magenta/tree/master/magenta/models/drums_rnn.

- Magenta, G. 2016b. Melodyrnn. https://github.com/tensorflow/magenta/tree/master/magenta/models/melody_rnn.
- Pachet, F. 2003. The continuator: Musical interaction with style. *Journal of New Music Research* 32(3):333–341.
- Roberts, A.; Engel, J.; Raffel, C.; Hawthorne, C.; and Eck, D. 2018. A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music. In *Proceedings of the International Conference on Machine Learning*.
- Rumelhart, D. E.; Hinton, G. E.; and Williams, R. J. 1986. Learning representations by back-propagating errors. *Nature* 323:533–.
- Simon, I., and Oore, S. 2017. Performance rnn: Generating music with expressive timing and dynamics. <https://magenta.tensorflow.org/performance-rnn>.
- Thom, B. 2000. Unsupervised Learning and Interactive Jazz/Blues Improvisation. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*.
- Thom, B. 2001. Machine learning techniques for real-time improvisational solo trading. In *Proceedings of the 2001 International Computer Music Conference*.
- Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017. Attention is All you Need. 5998–6008.
- Weinberg, G.; Malakarjuna, T.; ; and Raman, A. 2009. Interactive jamming with shimon: A social robotic musician. In *Proceedings of the ACM/IEEE International Conference on Human Robot Interaction*, 233–234.

Generative Design in Minecraft: Chronicle Challenge

Christoph Salge

School of Computer Science
University of Hertfordshire
Hatfield, United Kingdom
ChristophSalge@gmail.com

Christian Guckelsberger

School of Electronic Engineering and Computer Science
Queen Mary, University of London
London, United Kingdom
christian.guckelsberger@qmul.ac.uk

Michael Cerny Green

Game Innovation Lab
New York University
New York, USA
mcg520@nyu.edu

Rodrigo Canaan

Game Innovation Lab
New York University
New York, USA
rnc602@nyu.edu

Julian Togelius

Game Innovation Lab
New York University
New York, USA
julian@togelius.com

Abstract

We introduce the Chronicle Challenge as an optional addition to the Settlement Generation Challenge in Minecraft. One of the foci of the overall competition is adaptive procedural content generation (PCG), an arguably under-explored problem in computational creativity. In the base challenge, participants must generate new settlements that respond to and ideally interact with existing content in the world, such as the landscape or climate. The goal is to understand the underlying creative process, and to design better PCG systems. The Chronicle Challenge in particular focuses on the generation of a narrative based on the history of a generated settlement, expressed in natural language. We discuss the unique features of the Chronicle Challenge in comparison to other competitions, clarify the characteristics of a chronicle eligible for submission and describe the evaluation criteria. We furthermore draw on simulation-based approaches in computational storytelling as examples to how this challenge could be approached.

Introduction

In this paper we introduce the new *Chronicle Challenge* as additional, optional part of the *Generative Design in Minecraft (GDMC) Settlement Generation Competition*¹ (Salge et al., 2018). For the original competition, participants are required to submit code that creates a Minecraft (Persson, 2011) settlement in an unseen map. The goal is to foster interest in the problems of *adaptive* and *holistic* procedural content generation (PCG) (Shaker, Togelius, and Nelson, 2016; Compton, 2016; Short and Adams, 2017), and to provide a platform on which different solutions can be compared. Rather than starting from a blank slate, an *adaptive* generator must produce an artefact, i.e. a settlement, in response to existing content such as the map layout

¹Further information about the competition can be found on our website: <http://gendesignmc.engineering.nyu.edu/>

and climate. This adaptivity also sets this apart from other PCG challenges, such as the NaNoGenMo Cook and Colton (2018), where participants had to generate a novel. Furthermore, by *holistic* we mean that different types and aspects of content should fit well with each other, and potentially echo interactions in-between (Liapis, 2015; Liapis et al., 2019). For instance, a good entry would reflect how a settlement has been constrained and influenced by e.g. mountains and climate, but also how this settlement has shaped the surrounding landscape over time.

There are numerous examples of well-crafted human settlements that master these challenges, yet no human comparable AI solution exists. Eventually, we want to see generated settlements that are on par with human creations, and understand the underlying creative process. As in many other creative tasks, there is no well-defined, “optimal” solution that could be fully captured, or even approximated, by an objective function (cf. Smith, 2012, Ch. 8). This property characterizes many challenges in computational creativity (CC), distinguishing the field from general AI research (Colton and Wiggins, 2012). We can thus identify adaptive and holistic settlement generation as a CC challenge.

Given the vague nature of the objectives, the artefacts are judged by human referees based on the criteria of *Adaptivity*, *Functionality*, *Evocative Narrative* and *Aesthetics*. Each criterion is evaluated based on a list of illustrative questions. *Adaptivity* is defined as making a settlement that fits into the given map. In the previous challenge, participants dedicated a lot of work to appropriate building placement, and to generate buildings that reflect the existing natural resources. However, the current approaches feature very little “big picture” adaptation, e.g. algorithms do not yet decide whether a large farming village or a fortress would be the more appropriate settlement for a given map. *Adaptivity* (Lopes and Bidarra, 2011) is one of the core aspects of this challenge, and also permeates the other scoring categories. *Functionality* is defined as providing affordances

(Gibson, 1966) to hypothetical players and villagers. Here we benefit from the advantage that Minecraft is a game, and as such the world affords specific gameplay-relevant actions to the player (Cardona-Rivera and Young, 2013). Subsequently, structures in Minecraft can provide additional affordances relevant to player goals. Examples include bridges to allow for extra mobility, houses to protect from monsters, etc. The competition also considers affordances which have no direct relevance in the game, but would in reality. The criterion of *Aesthetics* is less about building a settlement that is beautiful, and more about avoiding errors that are obvious to humans, such as awkward placement or lack of proportion.

The first three criteria have been approached by past GDMC competition entries in a variety of ways (e.g. Brightmoore, 2018), but there has yet been little progress on creating an *Evocative Narrative*. The challenge here is to generate a settlement which, as an artefact, implicitly tells a story of how it came about, and of the people that inhabit it. Real-world settlements tell such stories, but also human-build settlements in Minecraft; people express cultural influences, and their settlements have an imagined or actual history that brought it about in a casual way. A settlement is the transient outcome of a creative process that usually involves many agents, and (computational) creativity researchers have long agreed that creativity does not happen in a vacuum (Jordanous, 2016). However, procedurally generated settlements often lack an Evocative Narrative.

To advance this aspect of our challenge we have therefore decided to introduce an optional bonus challenge, the Chronicle Competition, which adds the task of generating an explicit narrative, captured in natural language text. In the remainder of the paper, we outline the competition and discuss which unique challenges it offers in comparison to other benchmarks. To give participants a head start, we furthermore discuss how existing generative approaches could be applied to the challenge. We particularly consider approaches in *computational storytelling*, a CC subfield concerned with the study of algorithms capable of generating fictional narratives (Gervás, 2009; Berov, 2018). We thus connect this competition to existing work in CC, and yet open it up to researchers and the general public with an interest in PCG more generally.

The Chronicle Competition

The main task for the competition is to generate a *chronicle*, i.e. a written text about the history of a Minecraft settlement, and place it inside that settlement as a Minecraft book. We are deliberately vague about what exactly a chronicle is; to illustrate the range of encouraged submissions, we provide a number of examples. A chronicle could be a text written by different people during different times in the development of a settlement, or it could be a retelling of the town's history from a single, modern perspective, or even a tourist guide to historic buildings and places in the city. It can be written in different styles, and focus on different aspects, such as the lives of certain people, or buildings, or communities, etc. The chronicle can feature unreliable narrators and contradicting viewpoints. We explicitly encourage the use of focalization (Gervás, 2009), i.e. the restriction of what is

being told to what might have perceived by somebody in the scene. These examples are not exhaustive, and we encourage a wide variety of different submissions to delineate the scope further in the future. At this point, the only hard requirements are that the submission is in English and relates to specific generated settlements and how they came about.

Entries will be evaluated in terms of their (i) *Overall Quality*, and (ii) their *Fit* for a given settlement. For the evaluation of *Overall Quality*, we rely on the idea of producing objectivity by inter-subjectivity; each text will be evaluated by a number of human judges with diverse views. This is quite customary in competitions where humans or AIs generate creative game artefacts (Shaker et al., 2011; Stephenson et al., 2018; Khalifa et al., 2016, 2017). It alleviates the problem that there are no commonly accepted, computational measures for narrative quality. For example, some narratologists argue that a *story* is different to a *plot* or *narrative*, in that the earlier only represents a list of events, but the latter connects those with causal relations. Labov (1972) defines a minimal narrative as two states and a transition or movement in-between, where such a transition could be given by a causal relationship. However, others might consider causality as an aid in understanding a story, rather than as a strict requirement of a narrative (cf. Gervás, 2009). While there are some interesting metrics (e.g. Berov, 2017), automatic evaluation comes with the additional problem of elevating one or several specific metrics which then becomes the sole goals of optimization. Instead, we consequently encourage participants to use such metrics for the evaluation of their created artefacts, e.g. in “generate-and-test” approaches (cf. Togelius et al., 2011), but have human judges evaluate how well those criteria work. Since what makes a good narrative is still debated, we employ a range of soft constraints that are enforced through scoring. Rather than disqualifying a submission based on e.g. the absence of causality, we want to leave it to our judges to potentially give a lower score, but maybe also reward other, good features of the chronicle. The competition thus supports the discovery of new elements that make good narratives, and of evaluation criteria that could prove useful in computational storytelling, narratology and related fields. Our aim is to establish a relatively low hurdle for a minimally sufficient solution to encourage participation, but to also have a lot of room for improvement (Togelius, 2016).

The second criteria, *Fit*, will also be judged by humans, but is subject to more specific instructions. “Fit” is about how well a text corresponds to a specific settlement. Given that we usually have three competitions maps, imagine that a generator produces a settlement for each of those maps, and a chronicle for each settlement. Now imagine that we shuffle those chronicles around; would you still be able to assign each chronicle to the settlement it was originally generated for? Entries that show a clear relationship between the settlement and text would score high on Fit. This criterion inherits the focus of the overarching competition on adaptive PCG, for which generated content must be responsive to some other existing content. Ultimately, this touches upon an old problem of text generation, namely how to produce a text that is genuinely *about* something (Woods, 1981). We



Figure 1: Two examples of settlements from the 2nd GDMC competition, produced by different generators on the same map. To illustrate the idea of *fit* consider the following two chronicle fragments:

- 1) ... *we settled next to the desert tribe and placed a watchtower to keep an eye on them ...*
 - 2) ... *our village once was a trading post - with the rich traders' big houses cluttered around the central market square ...*
- For a high “Fit” score it should be evident which fragment belongs to which settlement.

think that it is possible to approach this without delving into the deep philosophical issues arising here, but nevertheless those issues are relevant. We believe that *Fit* has been neglected in computational storytelling so far, because existing systems mostly feature only one story world of limited complexity. Consequently, the generated narratives naturally fit into the bigger picture. Yet, we consider this an interesting challenge to inspire further developments in the domain, in particular for modular, service-based systems (León, 2011; Veale, 2013; Gervás, 2017) capable of generating narratives from exchangeable story worlds. In the Chronicle Competition, generators are evaluated on different maps, based on settlements with arbitrary complexity.

Possible Approaches

In this section we discuss possible approaches to chronicle generation based on existing work in computational storytelling, but we want to stress that there is no restriction in techniques for this competition; we explicitly want to encourage both *amateurs* and *specialized researchers* to find *new* solutions to the challenges involved. We highlight possible starting points, but also point out the challenges of the individual approaches for which this competition would offer an interesting and comparable benchmark. While there is a range of computational storytelling techniques, we focus mostly on those that can in one way or another deal with the holistic adaptation to existing content.

Recent data-driven approaches based on neural networks are capable of producing descriptions for photos (Donahue et al., 2015), or even to synthesize 3D scenes based on textual descriptions (Reed et al., 2016). The first technique could e.g. be used to generate data for storytelling from the perspective of the player character wandering through an existing settlement. The second technique could inversely generate a settlement from an existing chronicle. Unfortunately, these approaches usually require a lot of training data, which in the case of chronicles for Minecraft settlements is not available. It is also unclear if they would scale to the complexity required to tell a story about a whole set-

tlement. However, they might be useful in modular form, for example to generate the individual buildings of a settlement from a text description, to insert descriptions of buildings or natural sights into the chronicle, or to identify interesting elements in a settlement. There already is a model that can generate design descriptions for single buildings in Minecraft (Yoon et al., 2018). Existing approaches to generating text in specific styles could also be of interest, but they yet often struggle with adhering to a cohesive structure.

There are also a range of more structural approaches to computational narrative generation, which were surveyed by Gervás (2017) and Kybartas and Bidarra (2017), and can be split into roughly two categories. *Simulation-based approaches* simulate the interactions of agents and turning the recorded events into a narrative (e.g. Theune et al., 2003; León, 2011; Berov, 2018; y Pérez, 2015). A game-based example is the history generation of Dwarf Fortress (Hall, 2014), where generations of characters are born, fight, and die, producing a logbook of many events. For our competition, we might imagine some algorithm that successively builds a settlement and records events such as newly built houses, removing forests, etc. The problem with this naive approach though is that it misses out on establishing causal relationships between the events, and some might thus consider the resulting artefact only a basis for, but not an actual narrative. One popular means to overcome this in the cited work is to specify the beliefs and desires of the involved agents as source of meaningful, causal interaction.

The second category are *planning-based approaches* (e.g. Riedl and Young, 2010) which use propositional logic to generate narratives. Agents are be modelled with specific goals, and an ontology describes how certain actions produce certain outcomes. Planning is then used to determine which actions lead to the agent’s goals - giving each action a causal explanation. I.e. a settlement’s goal might be to have food production, and building farms might provide food production. In textual form this might lead to: “*We built farms on the slopes of the mountain to feed our people*”. The difficulty here is to design such an ontology in the first place

which fits well into the world, but this should generally be possible in a game such as Minecraft. With such an ontology, a planner could produce the narrative structure and simultaneously plan the settlement. It might then be worthwhile to add some noise to get a more exciting narrative. For instance, part of the settlement could burn down, an event that rational agents would not trigger as part of their plan.

Another issue with this kind of approach is to define the right kind of agent, with believable and interesting goals. The focal point of stories are mostly people, and their desires are relatable to us. A story about overcoming starvation or dangers is interesting, a story about an agent that wants to build 15 houses and then builds 15 houses is less so. Here the chronicle format might be a bit odd with most projects in computational storytelling which are very character focused. However, it is important to note that similarly, a settlement is shaped and experienced by characters; a city's population can be modelled as a single or multi-agent system with relatable human goals, such as, we felt threatened so we decided to build walls to protect us. Exemplary figures, such as the mayor, can serve as character and embody those views to give them a human perspective. Similarly, a multi-agent approach could simulate the interaction of different factions, each with their own goals, leading to conflicting actions. A trading guild might want to build a harbour, but the local farmers might sabotage this project because they fear competition with their crops, etc. Finally, a lot of these approaches can be combined. Given that participants can design both the settlement generator and the chronicle generator one approach would be to first design a process that simulates the causal chain of event that brings about a settlement. This can be done with a variety of simulated characters, ideally driven by believable motivations and encountering interesting random events. This, in essence, is very similar to what a lot of story generators do already. Then, this has to be followed by designing two projection functions, that translate this process a) into a textual history and b) into a 3d representation of the settlements. Both the settlement and the chronicle can be seen as imperfect projections, capturing different details, or a much richer actual history. Again, this is a problem not uncommon in computational storytelling, where several generators have a story graph that then gets translated into a text.

While we illustrated a breadth of existing approaches, we want to stress that a minimal solution to producing a chronicle could work with very simple techniques, like a text where placeholders are filled based on parameters derived from a settlement, such as “*We build a city in THE DESERT*”. At the same time, the challenges outlined here can be tackled with a lot of different, sophisticated methods and it would be interesting to see, if relying on them produces noticeable better results. We think that this challenge could serve as a platform to compare different approaches in a common task.

Future Plans

The Chronicle Challenge has once already been part of the annual GDMC competition. At present, both challenges are

interwoven, i.e. participants interested in chronicle generation must also provide a settlement generator, but not vice-versa. This allows for more freedom in the chronicle generation: a chronicle could be written post-hoc after the generation of a settlement based on the final artefact only, or alternatively as a settlement unfolds, leveraging a tight and unrestricted communication with the settlement generator, and potentially even interacting with it. The downside of this is that participants who are mainly interested in chronicle generation must also deal with the more general PCG challenges of settlement generation. As a consequence, the quality of chronicles may heavily depend on the quality of the settlement generator, and cannot be judged independently.

For future competitions we thus consider to offer a *standalone* Chronicle Challenge to attract more researchers from specialized fields such as computational storytelling. One option would be to ask participants to provide a chronicle generator for one specific Minecraft map with an existing settlement. However, this would be quite challenging in terms of extracting information about the settlement from a block-based representation. We may provide additional information such as building labels or historic events alongside the actual map, but this would require to first figure out what important information from settlement generation must be preserved for good chronicles. As an alternative option, we may give all participants one generator that creates a settlement over time and offers rich information along the way. All participants would thus have access to the same, rich time-sensitive data as input to their chronicle generator. For now, our plans are to rerun the chronicle competition as is and point interested participants to existing, open-source entries from previous years, that could be extended for chronicle generation.

Acknowledgments

CS is funded by the Marie Skłodowska-Curie grant 705643. CG is supported by EPSRC grant EP/L015846/1 (IGGI). RC gratefully acknowledges the financial support from Honda Research Institute Europe (HRI-EU). Many thanks to Leonid Berov and Dino Pozder for feedback on the initial idea for this competition through the lens of computational storytelling and narratology as part of the 2019 Dagstuhl seminar “Computational Creativity Meets Digital Literary Studies” (19172).

References

- Berov, L. 2017. Towards a computational measure of plot tellability. In *Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference*.
- Berov, L. 2018. A character focused iterative simulation approach to computational storytelling. In *International Conference on Interactive Digital Storytelling*, 494–497. Springer.
- Brightmoore, A. 2018. GDMC 2018 Competition - a participant's perspective.
- Cardona-Rivera, R. E., and Young, R. M. 2013. A cognitive theory of affordances for games. In *DiGRA Conference*.

- Colton, S., and Wiggins, G. A. 2012. Computational creativity: The final frontier? In *ECAI*, volume 12, 21–26. Montpellier.
- Compton, K. 2016. So you want to build a generator.
- Cook, M., and Colton, S. 2018. Neighbouring communities: Interaction, lessons and opportunities. In *International Conference on Computational Creativity*, 256–263.
- Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; and Darrell, T. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2625–2634.
- Gervás, P. 2009. Computational approaches to storytelling and creativity. *AI Magazine* 30(3):49–49.
- Gervás, P. 2017. Deconstructing computer poets: Making selected processes available as services. *Computational Intelligence* 33(1):3–31.
- Gibson, J. J. 1966. The senses considered as perceptual systems.
- Hall, C. 2014. Dwarf fortress will crush your cpu because creating history is hard. *Polygon*.
- Jordanous, A. 2016. Four perspectives on computational creativity in theory and in practice. *Connection Science* 28(2):194–216.
- Khalifa, A.; Perez-Liebana, D.; Lucas, S. M.; and Togelius, J. 2016. General video game level generation. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, 253–259. ACM.
- Khalifa, A.; Green, M. C.; Perez-Liebana, D.; and Togelius, J. 2017. General video game rule generation. In *Computational Intelligence and Games (CIG), 2017 IEEE Conference on*, 170–177. IEEE.
- Kybartas, B., and Bidarra, R. 2017. A survey on story generation techniques for authoring computational narratives. *IEEE Transactions on Computational Intelligence and AI in Games* 9(3):239–253.
- Labov, W. 1972. *Language in the inner city: Studies in the Black English vernacular*, volume 3. University of Pennsylvania Press.
- León, C. 2011. Stella—a story generation system for generic scenarios. In *Proceedings of the Second International Conference on Computational Creativity*.
- Liapis, A.; Yannakakis, G. N.; Nelson, M. J.; Preuss, M.; and Bidarra, R. 2019. Orchestrating game generation. *IEEE Transactions on Games* 11:48–68.
- Liapis, A. 2015. Creativity facet orchestration: the whys and the hows. In Lucas, S. M.; Mateas, M.; Preuss, M.; Spronck, P.; and Togelius, J., eds., *Artificial and Computational Intelligence in Games: Integration (Dagstuhl Seminar 15051)*, volume 5 of *I. Dagstuhl Reports*. 217.
- Lopes, R., and Bidarra, R. 2011. Adaptivity challenges in games and simulations: a survey. *IEEE Transactions on Computational Intelligence and AI in Games* 3(2):85–99.
- Persson, M. 2011. Minecraft.
- Reed, S.; Akata, Z.; Yan, X.; Logeswaran, L.; Schiele, B.; and Lee, H. 2016. Generative adversarial text to image synthesis. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48*, 1060–1069. JMLR. org.
- Riedl, M. O., and Young, R. M. 2010. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research* 39:217–268.
- Salge, C.; Green, M. C.; Canaan, R.; and Togelius, J. 2018. Generative design in Minecraft (GDMC): Settlement generation competition. In *Proceedings of the 13th International Conference on the Foundations of Digital Games*, FDG ’18, 49:1–49:10. New York, NY, USA: ACM.
- Shaker, N.; Togelius, J.; Yannakakis, G. N.; Weber, B.; Shimizu, T.; Hashiyama, T.; Sorenson, N.; Pasquier, P.; Mawhorter, P.; Takahashi, G.; et al. 2011. The 2010 mario ai championship: Level generation track. *IEEE Transactions on Computational Intelligence and AI in Games* 3(4):332–347.
- Shaker, N.; Togelius, J.; and Nelson, M. J. 2016. *Procedural content generation in games*. Springer.
- Short, T. X., and Adams, T. 2017. *Procedural Generation in Game Design*. CRC Press.
- Smith, A. M. 2012. *Mechanizing exploratory game design*. Ph.D. Dissertation, UC Santa Cruz.
- Stephenson, M. J. B.; Renz, J.; Ge, X.; Ferreira, L. N.; Togelius, J.; and Zhang, P. 2018. The 2017 aibirds level generation competition. *IEEE Transactions on Games*.
- Theune, M.; Faas, S.; Nijholt, A.; and Heylen, D. 2003. The virtual storyteller: Story creation by intelligent agents. In *Proceedings of the Technologies for Interactive Digital Storytelling and Entertainment (TIDSE) Conference*, volume 204215.
- Togelius, J.; Yannakakis, G. N.; Stanley, K. O.; and Browne, C. 2011. Search-based procedural content generation: A taxonomy and survey. *IEEE Transactions on Computational Intelligence and AI in Games* 3(3):172–186.
- Togelius, J. 2016. How to run a successful game-based ai competition. *IEEE Transactions on Computational Intelligence and AI in Games* 8(1):95–100.
- Veale, T. 2013. A service-oriented architecture for computational creativity. *Journal of Computing Science and Engineering* 7(3):159–167.
- Woods, W. 1981. Procedural semantics as a theory of meaning. In Joshi, A.; Webber, B.; and Sag, I., eds., *Elements of Discourse Understanding*. Cambridge, UK: Cambridge University Press. 300–334.
- y Pérez, R. P. 2015. A computer-based model for collaborative narrative generation. *Cognitive Systems Research* 36:30–48.
- Yoon, E.; Andersen, E.; Hariharan, B.; and Knepper, R. 2018. Design mining for minecraft architecture.

Shallow Art: Art Extension Through Simple Machine Learning

Kyle Robinson, Dan Brown
Cheriton School of Computer Science
University of Waterloo
Waterloo, ON N2L 3G1 Canada
{kyle.robinson, dan.brown}@uwaterloo.ca

Abstract

Shallow Art presents, implements, and tests the use of simple single-output classification and regression models for the purpose of art generation. Various machine learning algorithms are trained on collections of computer generated images, artworks from Vincent van Gogh, and artworks from Rembrandt van Rijn. These models are then provided half of an image and asked to complete the missing side. The resulting images are displayed, and we explore implications for computational creativity.

Introduction

The use of Machine Learning algorithms as assistive tools for the creation of visual and auditory artworks is an ever growing area of research (Magenta). In addition, machine learning methodologies—especially those based on neural networks—have been used in the creation of wholly new artworks (Gatys, Ecker, and Bethge 2016). The creation and use of such tools encourages discussions at the boundary between computational creativity and creativity support.

Many machine learning computational creativity applications have thus far used neural networks. While these methods have lead to extremely interesting and thought-provoking artwork (Klingemann; Obvious), their analysis from a creative perspective is complicated by an inability to interpret *why* the model acted as it did. This black-box effect can in some ways parallel our understanding of human creativity but differs in many ways, especially as neural networks are meticulously designed to fulfill their specific purpose.

In contrast to neural network based approaches of algorithmic art generation, Shallow Art applies more classical Machine Learning algorithms and methodologies to the problem of art generation. More specifically, Shallow Art applies computationally efficient classification and regression algorithms to the problem of art extension. We have developed a methodology that allows any single-output classification or regression algorithm to be trained on a dataset of images and then *complete* a partially provided image. We conclude by presenting outputs from various models trained using a number of different types of training images.

Related Work

Work on machine learning based creativity support is rich in breadth and depth. One significant source of recent research is the Magenta research division of Google Brain (Magenta). In the area of music generation, their work includes a model which creates piano performances featuring expressive changes in tempo and dynamics, as well as a musical counterpoint generator using a specifically-designed convolutional neural network (Oore et al. 2017; Huang et al. 2017). Additionally, Magenta tools transcribe polyphonic music, and synthesize sounds for music production (Hawthorne et al. 2018; Engel et al. 2019). In visual art, Magenta has developed a recurrent neural network able to create very simple stick drawings (Ha and Eck 2017). The network is trained on human drawings with a provided single classification and can generate drawings with or without a provided topic. These drawings are extremely simple in nature; consisting of just a few simple lines and shapes.

One interesting computational art generation research paper similar to Shallow Art attempts to learn the artistic style of one picture and transfer it to another as if the second image was created in the style of the first (Gatys, Ecker, and Bethge 2016). Where all of these research papers differ from Shallow Art is in their use of black-box or grey-box neural networks for training and generation. Conversely, Shallow Art prioritizes and enables the use of easily understandable and traceable machine learning algorithms such as decision trees and separating hyperplanes, with each pixel of the resultant project treated independently.

Methodology

Image Source

The data for this project is sourced through a combination of black-and-white computer-generated images of varying complexities, colored computer generated images, and a collection of artworks created by Vincent van Gogh and Rembrandt van Rijn. In the black-and-white data set each pixel of each image is either white or black. In the coloured image data sets each pixel is represented by a 3-tuple (*Red, Green, Blue*) of integers ranging in value from 0-255.

The computer generated image data sets were created using Python 3 and the Python Imaging Library through the

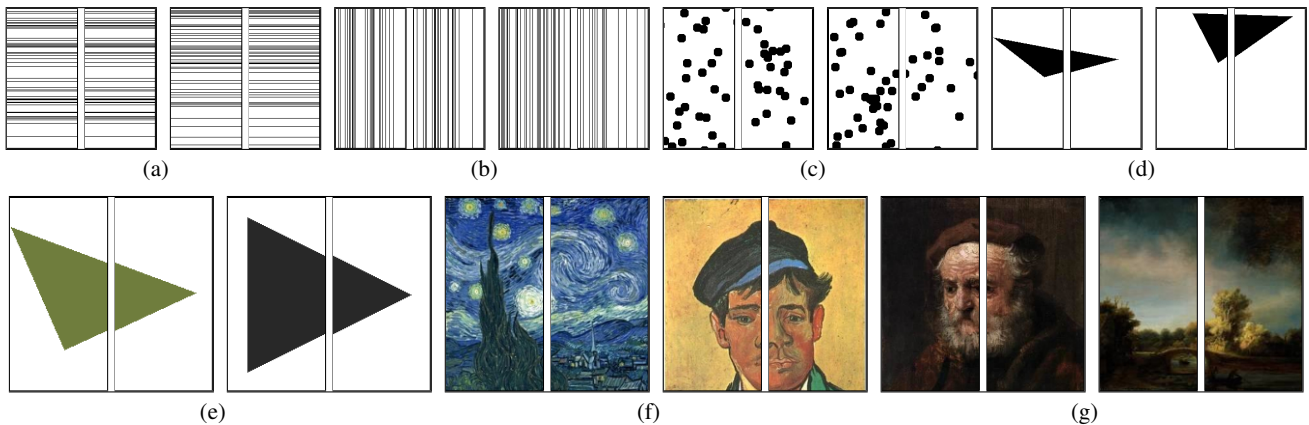


Figure 1: Image Data. Two examples each of a selection of image types used to train and test classification and regression models. (a)-(e) were randomly computer generated, whereas (f) and (g) are from collections of van Gogh and Rembrandt works respectively.

Pillow distribution (PIL). Four arbitrary patterns were selected for initial black and white image testing (figs. 1(a) to 1(d)): horizontal lines, vertical lines, triangles, and circles. Each image type eliminates all structure except one abstract concept which is exceedingly easy for a human to perceive. These generated images can also be sorted into two groups: those with unique solutions to the right half (that is, where knowing the left half implies the right half), and those without unique solutions to the right half. Note that as the resolution of the generated images decreases, the approximations of lines, and curves drastically decrease due to rasterization. Black and white images were generated at a resolution of 250x250 pixels and stored in the PNG format. Color images (including van Gogh and Rembrandt works) were generated or cropped to a resolution of 200x200 pixels due to limitations in the van Gogh and Rembrandt data repositories.

Lines Two types of line-based images were generated: horizontal, and vertical. Each image begins as a square white image, then horizontal or vertical lines one pixel thick are drawn at 50 random locations. Given any horizontal line image's left side the right side has only one correct solution, whereas the vertical line images have no left-to-right continuity. Examples of these images can be seen in Figures 1(a) and 1(b).

Circles Generated circle images consist of a white background with 50 randomly placed black circles. All 50 circles have a diameter of 15 pixels and may be placed at locations which clip with the edge of the image by up to their diameter. Circles are drawn as rasterized approximations of circles. Given any circle image's left side the right side contains some unique solutions (where circles are split in half) and many unknown solutions (the rest). Examples of circle images can be seen in Figure 1(c)

Triangles We generated triangle images with two randomly chosen vertices on the left side and one on the right. These vertices are then connected and filled in order to create a black triangle on a white background. Given any left-side image, predicting the right side is a relatively trivial task of

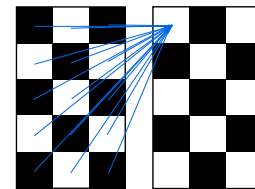


Figure 2: Each pixel on the right is predicted using the values of all pixels on the left (attributes). This process is repeated so that each pixel on the right half of the image is the output of one independent model.

extending the triangles' edges until they converge at the third vertex. Examples of these images can be seen in Figure 1(d). We also generated a set of triangle images that were each assigned a random fill colour and placed on a white background. Examples can be seen in Figure 1(e)

Van Gogh & Rembrandt We obtained Vincent van Gogh and Rembrandt van Rijn works (public domain) by scraping thumbnails from online galleries (Van Gough; Rembrandt). In total we collected 2,298 van Gogh images and 1,097 Rembrandt images. These images consist of finished and unfinished works of differing aspect ratios and sizes. We standardized these data sets by removing duplicates and non-rectangular images and scaling all remaining images to 200x200 pixels as seen in Figures 1(f) and 1(g). The van Gogh and Rembrandt images pixel data is stored and accessed in the 3-tuple RGB format described earlier.

Machine Learning

The task of binary prediction on the black and white data sets is one of binary classification, whereas the prediction of coloured images is one of regression, since the output RGB values must be between 0 and 255. In order to prepare images for machine learning classification and regression, each image is converted into a flattened one-dimensional array of values. For black and white images, each value is a binary representation of the pixel. For color images, each pixel cor-

responds to three values in the array, one for each colour shading. Image pixel arrays are split in half so that one half contains all pixels of the left side of the image and the other contains all pixels from the right side. The data array corresponding the left of the image is used as a series of attributes, and the array corresponding to the right side of the image is used as series of labels. In this way the black and white images contain $\frac{250 \times 250}{2} = 31,250$ training attributes each and the color images contain $\frac{(200 \times 200) \times 3}{2} = 60,000$ attributes each.

As traditional regression and classification tasks involve many attributes and one prediction per data point a method of converting single-output classification and regression models to multi-output is required. One simple and computationally efficient method of doing so is to train one model for each output required as described in Figure 2. Our implementation is altogether independent: it consumes any classification or regression model and trains one independent model for each of any number of required outputs. Due to the halving of images in this work the total number of trained models required for each image type is equal to the number of attributes. There are thus 31,250 models for black and white images and 60,000 models for color images. We refer to a collection of models which has been trained for per pixel image extension as a Wrapper-Model (WM). Importantly, WM's do not take into account any relationships (positional or otherwise) of the pixel data; each pixel model operates completely independently of all others within a WM. We isolated training and testing image data from each other, and implemented models using scikit Learn without hyperparameter optimization (Pedregosa et al. 2011).

In general, our WM implementation allows for the use of any classification/regression model, though computational feasibility is tightly interwoven with the computational complexity of any underlying model selected. We focused on easily interpretable models. Black and white image data sets were used to train four different types of WM's: decision tree, random forest, perceptron, and linear SVM. Decision tree and random forest are both tree-based algorithms capable of n -dimensional decision boundaries, whereas perceptron and linear SVM learn a linear separating hyperplane with a singular linear decision boundary. Only decision tree WM results are presented for colour images due to model training time constraints.

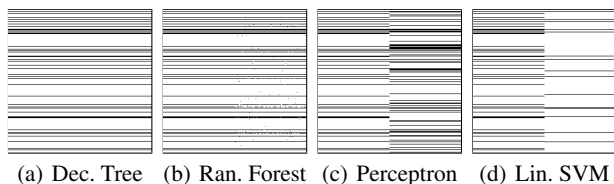


Figure 3: Horizontal Lines. Images created by classification Wrapper-Models trained on 50 randomly generated training images as described in Figure 1(a).

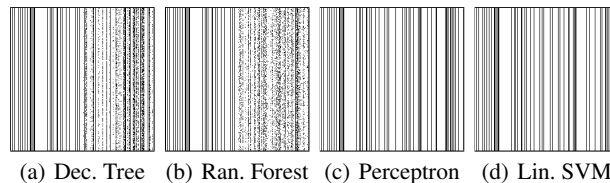


Figure 4: Vertical Lines. Images created by classification Wrapper-Models trained on 50 randomly generated training images as described in Figure 1(b).

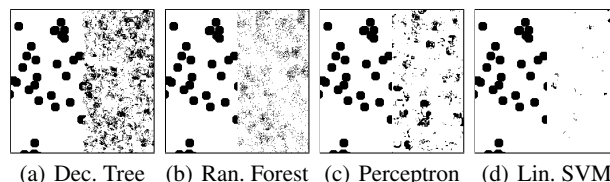


Figure 5: Circles. Images created by classification Wrapper-Models trained on 50 randomly generated training images as described in Figure 1(c).

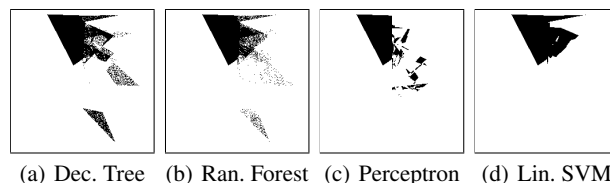


Figure 6: Triangle. Images created by classification Wrapper-Models trained on 50 randomly generated training images as described in Figure 1(d).

Results & Discussion

Black and White

Each black and white image type was used to train four WM's, one for each learning approach. Each WM was trained on 50 images¹ and then provided a single left side of the same image type and asked to predict the missing right side. Figures 3 to 6 present the outputs from each black and white WM for horizontal lines, vertical lines, circles, and triangles respectfully. The left half of each image is the raw input image and the right half is the output prediction from the WM.

Both tree-based WM's were able to correctly predict the horizontal line images (figs. 3(a) and 3(b)), whereas the separating hyperplane WM's were not (figs. 3(c) and 3(d)). The opposite is true for the vertical line images where perceptron and linear SVM algorithm outputs look much more natural (fig. 4). The accuracy on the vertical line images is far less important than the continuity of the lines which are drawn. Outputs from the circle WM show that none of the models was able to learn the general pattern of circles with the limited number of training images, but each algorithms attempt looks different (fig. 5). The triangle image WM's all show a general inclination towards convergence, though they differ

¹Black and white images were used as a proof of concept and were therefore trained on small data sets.

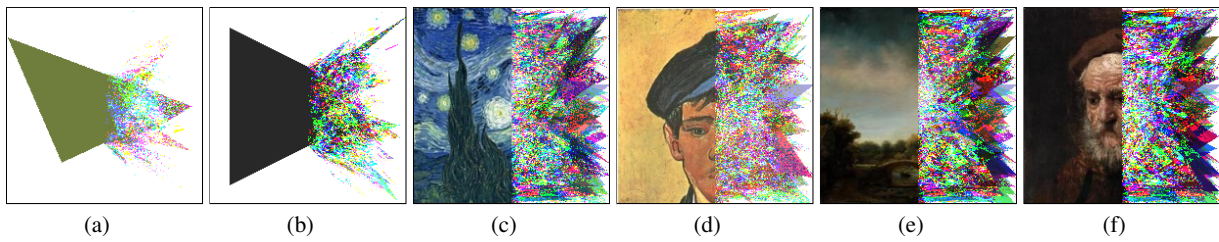


Figure 7: Triangle WM. Outputs from a decision tree WM trained on 1900 images of triangles as described in Figure 1(e)

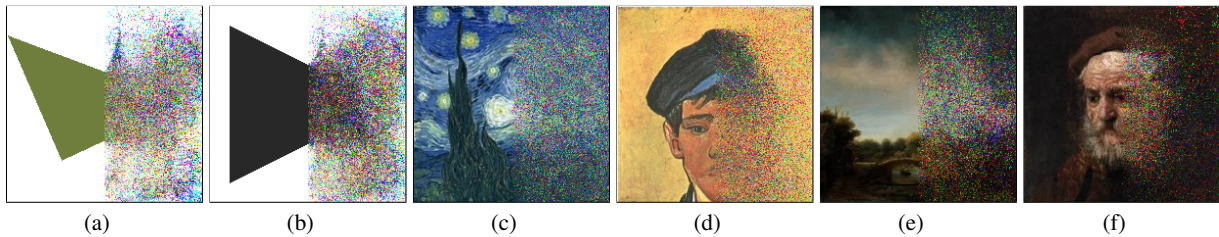


Figure 8: Vincent van Gogh WM. Outputs from a decision tree WM trained on 1900 van Gogh works as described in Figure 1(f)

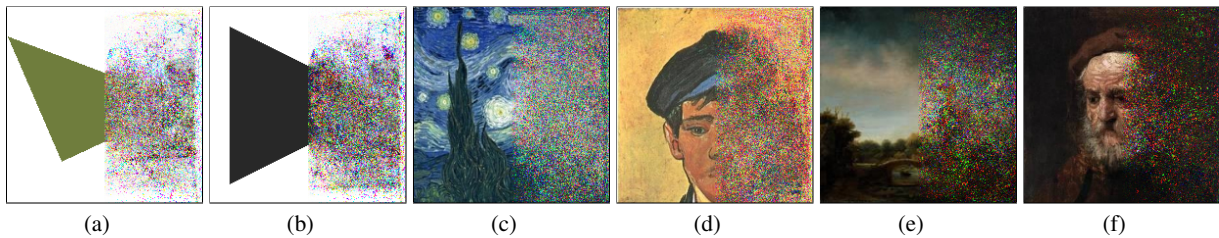


Figure 9: Rembrandt van Rijn WM. Outputs from a decision tree WM trained on 900 Rembrandt works as described in Figure 1(g)

drastically in their *interpretation* of how to complete the image. More testing with larger training data sets is required to fully interpret how and if the models converge.

Colour

Colour image types were each used to train a single decision tree WM and then used to predict two images of every other colour image type. For example, a decision tree WM was trained on van Gogh images and then provided two image halves from the coloured triangle, van Gogh, and Rembrandt data sets each to extend. In this sense each model is trained on one *style* and then asked to complete images that come from other *styles*. The colored triangle and van Gogh models were trained using 1900 images, and the Rembrandt model was trained using 900 images. Figures 7 to 9 present the outputs from each of the three trained colour decision tree WM's. As with the black and white images, the left half is the raw input image and the right half is the WM's output.

In all cases each model shows that it is able to predict the general colours and shapes of images in the same style despite never having seen them before. The green triangle WM image shows convergence towards a single point, whereas the second from the left image shows a large spectrum of dark colours with no clear convergence (figs. 7(a) and 7(b)). When tasked with extending van Gogh and Rembrandt works the triangle WM outputs are expectedly abstract (figs. 7(c) to 7(f)). Van Gogh and Rembrandt WM's are both able to complete images in the style of each other

while maintaining a sense of colour and shape; this is especially seen in Figures 8(d), 8(f), 9(d) and 9(f). When posed with the abstract triangle images the van Gogh and Rembrandt models correctly paint the white background on the fringes and in Figure 8(b) even extend the dark colours from the triangle.

Conclusion

Through the implementation of a wrapper method which can utilize any single-output classification or regression model to learn from and complete images, Shallow Art presents a novel new method of computational art generation which straddles the boundary between computational creativity and creativity support.

Shallow Art differs from previous approaches in computational creativity support and art generation in its focus on interpretability. This interpretability paves the way for a new analytical perspective which neural network based approaches do not provide. In addition, when framed as an agent in a co-creative process Shallow Art presents an interesting new perspective to discussion on co-creativity, and might enable multi-round experimentation between a human and the Shallow Art system.

Using abstract computer-generated visuals in tandem with artworks from famous classical artists as training data for simple machine learning methods has opened the door to future experimentation and analysis on the topic of non-neural network based approaches to computational art creation.

References

- Engel, J.; Agrawal, K. K.; Chen, S.; Gulrajani, I.; Donahue, C.; and Roberts, A. 2019. GANSynth: Adversarial Neural Audio Synthesis. *CoRR* abs/1902.08710.
- Gatys, L.; Ecker, A.; and Bethge, M. 2016. A neural algorithm of artistic style. *Journal of Vision* 16(12):326.
- van Gough, V. Vincent van Gogh - The Complete Works. <https://www.vincent-van-gogh-gallery.org/>. Accessed: 2019-05-01.
- Ha, D., and Eck, D. 2017. A Neural Representation of Sketch Drawings. *CoRR* abs/1704.03477.
- Hawthorne, C.; Elsen, E.; Song, J.; Roberts, A.; Simon, I.; Raffel, C.; Engel, J.; Oore, S.; and Eck, D. 2018. Onsets and Frames: Dual-Objective Piano Transcription. In *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, 2018*.
- Huang, C. A.; Cooijmans, T.; Roberts, A.; Courville, A. C.; and Eck, D. 2017. Counterpoint by Convolution. In *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, 211–218.
- Klingemann, M. Quasimondo. <http://quasimondo.com/>. Accessed: 2019-05-01.
- Magenta. Magenta - Google AI. <https://magenta.tensorflow.org/>. Accessed: 2019-05-01.
- Obvious. Obvious - Gallery. <http://obvious-art.com/gallery.html>. Accessed: 2019-05-01.
- Oore, S.; Simon, I.; Dieleman, S.; and Eck, D. 2017. Learning to Create Piano Performances. In *NIPS 2017 Workshop on Machine Learning and Creativity*, https://nips2017creativity.github.io/doc/Learning_Piano.pdf.
- Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12:2825–2830.
- PIL. Pillow. <https://python-pillow.org/>.
- Rembrandt. Rembrandt - The Complete Works. <https://www.rembrandtonline.org/>. Accessed: 2019-05-01.

Engagement-Reflection in Software Construction

Quinten Rosseel and Geraint A. Wiggins

Artificial Intelligence Lab, Vrije Universiteit Brussel, Pleinlaan 9, 1050 Brussels, Belgium
quinten.rosseel@vub.be, geraint@ai.vub.ac.be

Abstract

This paper introduces experimental research in progress concerning a novel creative software construction method that uses the Engagement-Reflection model and Floyd-Hoare logic. By considering software construction as a creative writing task, we demonstrate how principles from story generation can be applied in software construction.

Introduction

In this paper, we present work in progress on the automated construction of programs. Our approach differs from most such activities because we view program construction as a creative process, rather than a deductive one, and we apply technology from computational creativity to address it. The Engagement-Reflection model (Sharples, 1995) is a well-known approach to narrative generation. Here, we metaphorically replace the narrative with a program, and the well-formedness constraints on narratives with program semantics specified using a standard method of program analysis.

The paper is laid out as follows. In the background section, we illustrate prior research in creative story writing and story generation that inspired this work. More specifically, the Engagement-Reflection model (Sharples, 1995) and an application of it, named MEXICA (Pérez y Pérez and Sharples, 2001).

In the preamble, we look more closely at Floyd-Hoare (FH) logic (Hoare, 1969), the principal formalism for reasoning about programs in this work. We employ FH-logic for describing assignments with boolean conditions over program variables and extend it with distance measures and transformations that enable construction of simple programs within the ER-model, given user-defined specifications.

After a technical description of the construction process, we provide an outlook for the most important challenges that are to be tackled in the research following this paper.

In essence, the conditions set out by the user define the conceptual space (Boden, 1992) of programs that the system can explore. Exploration of this conceptual space is done by virtue of the ER-cycle that drives program composition forward. After each ER-cycle, the conditions bounding the conceptual space can be transformed by applying transfor-

mational rules, yielding new conceptual spaces that can in turn be explored by additional ER-cycles.

Background

The Engagement-Reflection (ER) Model

The ER-model of Sharples (1995) is a cognitive model that describes the creative process of authors writing stories. The idea is to decompose the writing task into two phases called engagement and reflection. This composition is based on the observation that a writer cannot simultaneously enact a writing procedure and re-represent it at the same time (Karmiloff-Smith 1990, quoted by Sharples 1995). This means that reflecting on a text requires the writer to stop writing, resulting in a cycle of engaged knowledge telling, interleaved with periods of reflection (Sharples, 1995).

Cycles can have a short period, as when a writer looks back over each sentence as it is written, or a longer period, as when a writer looks back over a paragraph as it is written. The interaction between engagement and reflection pushes the composition of material forward, with engagement providing new material for consideration and reflection offering a re-interpretation of the material, together with new plans to be enacted (Sharples, 1995).

The ER-model has been successfully applied in story generator MEXICA (Pérez y Pérez and Sharples, 2001), that produces story frameworks about the Mexicas, old inhabitants of modern México City. MEXICA shows that it is possible to generate coherent content that satisfies an initial set of user-defined constraints using the ER-model.

From Stories to Programs

Cognitive models of creative writing and story generators are of interest because stories and programs share properties that suggest how ideas from creative writing procedures can be applied in the domain of software construction.

Characters and variables are arguably the most straightforward correspondence. Just as characters may take different forms and personalities, variables may be bound to different types and values. Characters in the story are subject to actions and events, pushing the emerging story in a particular direction. Likewise, variables are subject to statements calls that modify the variable state, causing the program to behave accordingly.

In most stories, the plot builds up to one or more protagonists that partake in some final activity or event that ends the story. In the same way, the goal of the program is to assign a set of variables to a desired value or mutual relation. In some cases, story actions and events require that characters are in a particular state or mutual relation before they can be executed. This accords with conditional statements of the programming language.

Preamble

Floyd-Hoare (FH) Logic

FH logic is the principal formalism behind the software construction processes in this work. The central component of the logic is the Hoare triple, that represents how a set of statements \mathcal{S} changes the variable state. This state is captured in a set of conditions on the values of the variables.

Condition A condition c is a boolean assertion on the value of a variable, before or after execution of a set of statements \mathcal{S} . Conditions are stored in a global set of conditions \mathcal{C} , statements are stored in a global set of statements \mathcal{S} .

Hoare Triple $(Q) \mathcal{S} (\mathcal{R})$ is a Hoare triple that asserts that an ordered set of program statements $\mathcal{S} \subseteq \mathcal{P}(\mathcal{S})$, satisfying a set of preconditions $Q \subseteq \mathcal{P}(\mathcal{C})$ before running, will satisfy a set of post-conditions $\mathcal{R} \subseteq \mathcal{P}(\mathcal{C})$ after running. All Hoare triples are stored in the global set of Hoare triples \mathcal{T} . \mathcal{P} is the standard power set operation over a set.

In practice, we use triple constructions backwards when generating programs. We know that the result of a set of statements \mathcal{S} satisfies a program state \mathcal{R} , and we need to infer an assertion Q on the state before \mathcal{S} , provided that \mathcal{S} terminates.

Axiom of Assignment When we know that a set of conditions Q is true after assigning an expression E to variable V , then this means that the substitution of V by E in the set of conditions Q must hold before the assignment. Note that $Q \subseteq \mathcal{P}(\mathcal{C})$ and $\{(V := E)\} \subseteq \mathcal{P}(\mathcal{S})$.

$$\frac{}{\overline{(Q_{[V \mapsto E]}) \{(V := E)\} (Q)}}$$

The assignment axiom has no premises and generates the weakest set of preconditions that is needed for the execution of the assignment to result in a state that satisfies the set of post-conditions (Bridge, 2003).

Example 1 Proof that $x = 1$, given that $x = 2$ after the assignment $x := x + x$ using the assignment axiom.

$$\frac{\frac{\overline{(\{x = 2\}_{[x \mapsto (x+x)])} \ x := (x+x) \ (\{x = 2\})}}{\overline{(\{x+x = 2\}) \ x := (x+x) \ (\{x = 2\})}}}{\overline{(\{x = 1\}) \ x := (x+x) \ (\{x = 2\})}}$$

□

Extensions to Floyd-Hoare logic

If we know how to transform a program state \mathcal{R} to another state Q with a set of statements \mathcal{S} , example 1 suggests that

program construction is possible with a set of start conditions $Q \subseteq \mathcal{P}(\mathcal{C})$, a set of goal conditions $\mathcal{R} \subseteq \mathcal{P}(\mathcal{C})$ and an ordered set of statements $\mathcal{S} \subseteq \mathcal{P}(\mathcal{S})$.

Valued Condition Distance Inserting relevant statements requires a distance metric that quantifies how a set of statements influences the state of a program with respect to its goal. When conditions are real-valued, a distance δ between two conditions sets $Q \subseteq \mathcal{P}(\mathcal{C})$ and $\mathcal{R} \subseteq \mathcal{P}(\mathcal{C})$ can be determined by a pairwise difference of each variable v , asserted in both Q and \mathcal{R} , where $E(c)$ evaluates a condition $c \in \mathcal{C}$ to the asserted value of its variable.

$$\delta(Q, \mathcal{R}) = \sum_{\forall v: q_v \in Q, r_v \in \mathcal{R}} |E(q_v) - E(r_v)|$$

Condition Variables In order to specify abstract relations over triple conditions, we introduce condition variables, stored in a global dictionary \mathcal{V} . Each entry (C, p) in \mathcal{V} is a one-to-one mapping between a condition variable C , denoted with an uppercase letter, to a program variable p , denoted with a lowercase letter. Condition variables enable us to specify abstract relations that allow conditions to satisfy a broader set of program states.

Abstract Condition Distance Conditions that involve abstract expressions cannot be evaluated by value but are compared with respect to an abstract distance function. In this context, we look at the minimum tree edit distance (TED) of the abstract syntax tree (AST) associated with the expression in the condition. Hence, insertions, deletions and substitutions are executed on the level of operators, function applications, variables and values.

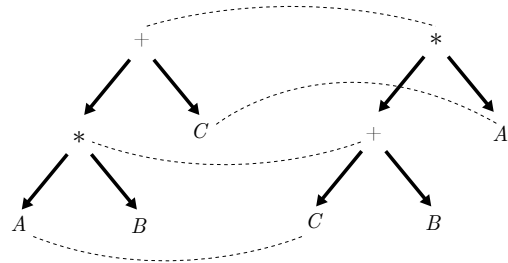


Figure 1: A mapping between $A * B + C$ and $(C + B) * A$ with 4 substitutions.

The TED problem has a well-known, dynamic programming solution (Zhang and Shasha, 1989) that allows for a customizable cost function γ to evaluate insertions, deletions and substitutions of nodes in trees. This is desirable as not all tree operations need the same associated weight in the distance metric.

Variable Dependency Transformation Being informed on condition variables in \mathcal{V} , enables the system to rewrite conditions from triples in \mathcal{C} when given patterns arise.

The following rule is aimed at separating abstract variable dependencies. A Hoare triple with a set of statements $\mathcal{S} \in$

$\mathcal{P}(\mathcal{S})$, a set of post-conditions $\mathcal{R} \subseteq \mathcal{P}(\mathcal{C})$ and a singleton precondition of the form

$$\left\{ (p_0 = f_1(C_1) \circ_1 \dots \circ_{n-1} f_n(C_n)) \right\} \subseteq \mathcal{P}(\mathcal{C})$$

with $f_1 \dots f_n$ functions involving one and only one condition variable, $C_0 \dots C_n$ mutually independent condition variables and $\circ_1 \dots \circ_{n-1}$ binary operations $\in \{\oplus, \ominus, \otimes, \odot\}$, can be transformed with the variable dependency transformation when $(C_0, p_0) \dots (C_n, p_n) \in \mathcal{V}$.

$$\frac{\left(\left\{ (p_0 = f_1(C_1) \circ_1 \dots \circ_{n-1} f_n(C_n)) \right\} \right) S(\mathcal{R})}{\left(\left\{ (p_1 = f_1(C_1)) \dots (p_n = f_n(C_n)) \right\} \right) p_0 = p_1 \circ_1 p_2 \dots p_{n-1} \circ_{n-1} p_n; S(\mathcal{R})}$$

Example 2 The following proof employs condition variables and a variable dependency transformation. The goal is to generate the Pythagoras relation between three program variables a, b and c . The start specifications t_1, t_2 and t_3 make sure that $(A, a), (B, b)$ and $(C, c) \in \mathcal{V}$.

$$t_1 : \left(\emptyset \right) \emptyset \left(\left\{ (a = A) \right\} \right)$$

$$t_2 : \left(\emptyset \right) \emptyset \left(\left\{ (b = B) \right\} \right)$$

$$t_3 : \left(\emptyset \right) \emptyset \left(\left\{ (c = C) \right\} \right)$$

The goal specification t_4 is defined in terms of A and B and should be satisfied for all programs that are generated. By equaling c to both C and $\sqrt{A^2 + B^2}$ in the start and goal conditions, the system is told to find a way to make $C = \sqrt{A^2 + B^2}$.

$$t_4 : \left(\left\{ (c = \sqrt{A^2 + B^2}) \right\} \right) \emptyset \left(\emptyset \right)$$

As the derivation process works backwards from t_4 , the system might at some point infer a triple t_5 using the assignment axiom. Assume that c is positive in this example.

$$\frac{\left(\left\{ (c = \sqrt{A^2 + B^2})_{[c \rightarrow \text{sqrt}(c)]} \right\} \right) c := \text{sqrt}(c); \left(\left\{ (c = \sqrt{A^2 + B^2}) \right\} \right)}{\left(\left\{ (\sqrt{c} = \sqrt{A^2 + B^2}) \right\} \right) c := \text{sqrt}(c); \left(\left\{ (c = \sqrt{A^2 + B^2}) \right\} \right)}$$

$$t_5 : \left(\left\{ (c = A^2 + B^2) \right\} \right) c := \text{sqrt}(c); \left(\left\{ (c = \sqrt{A^2 + B^2}) \right\} \right)$$

Using the variable dependency transformation rule, the system is now able to rewrite triple t_5 as follows. This allows the triple to be further evaluated in terms of A and B separately.

$$\frac{t_5 : \left(\left\{ (c = A^2 + B^2) \right\} \right) c := \text{sqrt}(c); \left(\left\{ (c = \sqrt{A^2 + B^2}) \right\} \right)}{\left(\left\{ (a = A^2), (b = B^2) \right\} \right) c := a + b; c := \text{sqrt}(c); \left(\left\{ (c = \sqrt{A^2 + B^2}) \right\} \right)}$$

Finally, in the optimal case, the system might finish the program by applications of the assignment axiom, resulting in t_6 and t_7 .

$$\frac{\left(\left\{ (a = A^2)_{[a \rightarrow (a*a)]}, (b = B^2)_{[b \rightarrow (a*a)]} \right\} \right) a := a * a; \left(\left\{ (a = A^2), (b = B^2) \right\} \right)}{t_6 : \left(\left\{ (a = A), (b = B^2) \right\} \right) a := a * a; \left(\left\{ (a = A^2), (b = B^2) \right\} \right)}$$

$$\frac{\left(\left\{ (a = A)_{[b \rightarrow (b*b)]}, (b = B^2)_{[b \rightarrow (b*b)]} \right\} \right) b := b * b; \left(\left\{ (a = A), (b = B^2) \right\} \right)}{t_7 : \left(\left\{ (a = A), (b = B) \right\} \right) b := b * b; \left(\left\{ (a = A), (b = B^2) \right\} \right)}$$

□

Construction of Programs

This section provides more background on how program construction employs previously described concepts to generate software in the C programming language. We distinguish three main processes at the highest level.

- Program Input Parsing: extract and generalize statements from the input programs to Hoare triples.
- Program Sequence Engagement: select Hoare triples and construct programs.
- Program Sequence Reflection: verify and edit engaged program sequences and its Hoare triples according to the specifications set out by the user.

In order to generate programs, the user provides an inventory of programs and specifies the start and goal conditions in the form of Hoare triples, like for example t_1, t_2, t_3 and t_4 in example 2. After parsing the inventory, the system uniformly draws statements from \mathcal{S} and retains those that reduce the distance between the user-defined start and goal conditions.

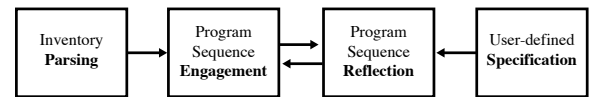


Figure 2: The program construction process.

Program Input Parsing

Before any meaningful program can be constructed, the system requires an inventory of valid C programs to serve as experience during engagement and reflection. This inventory is parsed and represented as Hoare triples in the triple store \mathcal{T} together with statements in the statement store \mathcal{S} .

At parse-time, statements are generalized before storage in \mathcal{S} so that they can be instantiated to a variable that is part of the user's specification. Additionally, user-defined goal specifications are not taken into account at parse-time as boolean assertions inside conditions are only initialized upon reaching reflection. However, each statement inside

a Hoare triple gets equipped with a Hoare transition rule that semantically corresponds with that statement, such that boolean assertions can get propagated through when they are used inside triples.

It is worth mentioning that there are many ways to extract triples from an inventory of programs. We choose to retain the semantics of the inventory as much as possible when storing triples inside the triple store \mathcal{T} and do not bother to extract every possible sub-program within a parsed program, as this would cause a combinatorial explosion of triples in \mathcal{T} . Although we see that it can yield benefits for better and more diverse composition, this is not the current focus in this research.

Program Sequence Engagement

The engagement phase constructs programs from the knowledge stored in the triple store \mathcal{T} , the statement store \mathcal{S} and the condition store \mathcal{C} . The selection process selects a random triple from \mathcal{T} and instantiates the triple's statements with a random variable from the user's specification in \mathcal{C} . Subsequently, triples are inserted to the program sequence, a structure that represent the program under construction. This process repeats until the maximum number of engagement iterations is reached.

We are aware of the fact that this process has more potential beyond random selection methods and that more informed selection procedures can be explored in future work.

Program Sequence Reflection

Reflection is the most decisive factor in the software construction process. When engagement transmits a program sequence, it's up to reflection to determine what triples are relevant for the specification passed by the user. Reflection picks up from the last analyzed triple in the program sequence and propagates its initialized post-conditions upwards through the triple's associated statements, using the Hoare transition rules. When a condition has propagated through all statements inside the triple, the condition is part of the set of preconditions of that triple. The goal is to obtain a triple with the same preconditions as the start conditions set out by the user. Reflection continues until the maximum amount of reflection cycles is reached or when the user-defined start conditions are satisfied.

A triple under consideration is deemed relevant when the list of statements inside that triple transforms the preconditions of the last analyzed triple closer to the start conditions. This is done by initializing the post-conditions of the triple under consideration, applying its transition rules and comparing these preconditions with the start conditions. If relevant, the triple is kept in the program sequence, otherwise it is removed.

One might rightfully object that this can introduce local minima with respect to the distance metric in the trajectory of triple insertions. Inserting one triple might not improve the conditions with respect to the start conditions, but it might improve the conditions as a combination with other triples. This leads into territory of Hoare triple merging and block structures, the current focus of this research and one of the topics in the next section.

Finally, when the maximum number of engagement iterations is reached, the system checks if it is able to rewrite the triples in \mathcal{T} by applying transformation rules that are triggered by predefined patterns in conditions, like the variable dependency transformation.

Conclusions

At this point, we have discussed the most significant mechanisms that underlie the program construction process of this work. This last section provides a short framing of the future challenges to be tackled.

Blocks and Control Flow

The system described up until now focuses on combinations of single statements, allowing it to compose simple programs in the C programming language. The next logical steps lie in the ability to work with compound statements or blocks, paving the way for control flow structures that specify the logical order in which computations are performed, like conditionals and loops.

Blocks are delimited by curly braces to group together declarations and statements, which makes them syntactically equivalent to single statements for the parser of a C compiler (Kernighan and Ritchie, 1978). All control-flow structures of interest employ block structures, so it is no surprise that Hoare triples need to be general and flexible in their representation for block structures. Additionally, block representations will need to be editable, such that reflection can evaluate and edit nested block statements too. Allowing block-structure flexibility will enable the system to combine and reuse different parts from the inventory of programs. These merging strategies will prove useful when local optima arise in the construction of programs.

On a more technical note, compound statements require the parser to pass additional information to the triple representation in \mathcal{T} so that its transitional rules can be applied accordingly. As each block deals with its own scope, the engagement phase needs to employ more informed initializations for the variables in each block and take into account lexical scoping.

Software Construction as a Creative Activity

Before we can consider the system creative, it needs a component that evaluates generated concepts (Wiggins, 2006). Currently, system output is manually evaluated after the start conditions are satisfied by the ER-cycle. In future work, more automated evaluation processes based on software engineering metrics of size and complexity need to be incorporated (Fenton and Bieman, 2014). This information can be used as a feedback signal to adapt the conceptual space that determines program construction, similar to MEXICA's filter system.

References

- Boden, M. 1992. *The Creative Mind*. London: Abacus.
- Bridge, D. 2003. Lecture 17: Floyd-hoare logic for partial correctness.

- Fenton, N., and Bieman, J. 2014. *Software metrics: a rigorous and practical approach*. CRC press.
- Hoare, C. A. R. 1969. An axiomatic basis for computer programming. *Communications of the ACM* 12(10):576–580.
- Karmiloff-Smith, A. 1990. Constraints on representational change: Evidence from children’s drawing. *Cognition* 34(1):57–83.
- Kernighan, B. W., and Ritchie, D. M. 1978. *The C programming language*. Prentice Hall.
- Pérez y Pérez, R., and Sharples, M. 2001. Mexica: A computer model of a cognitive account of creative writing. *Journal of Experimental & Theoretical Artificial Intelligence* 13(2):119–139.
- Sharples, M. 1995. *An account of writing as creative design*. University of Sussex.
- Wiggins, G. A. 2006. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems* 19(7):449–458.
- Zhang, K., and Shasha, D. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM journal on computing* 18(6):1245–1262.

Modeling Knowledge, Expression, and Aesthetics via Sensory Grounding

Brad Spendlove and Dan Ventura

Computer Science Department

Brigham Young University

Provo, UT 84602 USA

brad.spendlove@byu.edu, ventura@cs.byu.edu

Abstract

In many instances, computationally creative systems need to meaningfully interact with human agents—by existing in the environment they live in and/or interfacing with one or more forms of human perception and/or understanding the way that humans structure their perceptions of the environment. In this preliminary work, we present a model of a mind’s knowledge base consisting of concepts that are grounded in sensory perception of environmental phenomena. Although the individual components of this model are complex and difficult to fully define, our model’s structure describes how knowledge is acquired and used for expression—including creative expression—and aesthetic assessment. Our formalization aims to serve as an abstract modularization that effectively factorizes these complex operations.

Introduction

The field of computational creativity includes the goal of developing increasingly powerful autonomous systems that can be considered truly creative in their own right. Achieving that goal is challenging in part due to the deep, inherent differences between computers and humans. Wiggins offers the following definition of creativity: “The performance of tasks which, if performed by a human, would be deemed creative” (2006). Implicit in this definition is the requirement that any agent, human or non-human, must engage with creativity on human terms¹.

This means that creative endeavors must exist within the scope of not only the environment in which humans exist but also the particular ways that environment is perceived by humans. Furthermore, creative performances presented for comparison to human creativity must be compatible with the way human thought is organized.

These factors indicate that the following related mechanisms may be useful inclusions in a model of computational creativity: a model of an environment, a model of how minds

¹Although later work, such as that by Colton and Wiggins, has defined creativity in a way that omits direct deference to humanity (2012), we argue that the now somewhat unfashionable human-centric definition is still relevant to an important subset of unbounded trans-human creativity, namely that subset that relates to humans in any way. As such, we are content with limiting our scope to that definition in this work.

perceive that environment, and a model of how minds structure those perceptions. We will refer to the quantization of environment as *phenomena*, the quantization of structured perceptions of phenomena as *concepts*, and the structure of all concepts in a mind as *knowledge*.

Modeling knowledge is challenging, but the lack of such knowledge in a computational model leaves a vacuum which is easy to underestimate. Identifying that vacuum and filling it with a useful knowledge model appears to be a fruitful avenue toward improving computational models of creativity.

Current computational knowledge bases such as ConceptNet (Speer and Havasi 2012), WordNet (Miller 1995), Datamuse², and word2vec (Mikolov et al. 2013) have proven useful for certain language-based tasks but fall short of a complete knowledge model due, at least in part, to a lack of grounding of the concepts contained therein.

Similar to how the words in a dictionary are defined exclusively in terms of other words, existing knowledge bases contain concepts and relations between them but are missing a crucial element that grounds those concepts’ meanings and relationships. Their concepts exist as if in a cloud, floating above the ground and impossible for a computer to align with the real world (at least as perceived by humans) without additional information.

Grounding provides intrinsic meaning that is independent from any observer or other knowledge. While not all concepts must be directly grounded, ensuring that all concepts are at least indirectly grounded (i.e., related to or built up from grounded concepts) will aid in attaining these properties of intrinsic, independent meaning.

We propose that perception—sensory experiences that arise from a mind observing its environment—is a useful basis upon which to ground concepts. Because the environment is consistent and exists independently from any observer, it can be relied upon to serve as a foundation for inherently meaningful concepts and knowledge.

We present a model that formalizes the acquisition of a knowledge base of concepts which are grounded in the sensory experiences that arise when a mind observes phenomena in its environment. This knowledge base may be employed by many faculties of that mind including expression of concepts as generated phenomena and aesthetic assess-

²<http://www.datamuse.com/api/>

ment of observed phenomena. While the ultimate goal of this model is to provide a framework for improved models of computational creativity, we expect that it may have application in other disciplines as well.

Our model could be useful to the field of computational creativity by serving either as a framework for the creation of computational minds or to approximately model human minds as part of a computational system, for example in order to predict how humans would react to a given creative expression. Although each element of our model is in itself complex and difficult to implement, we hope that our formalization allows an abstract modularization that effectively factorizes the problem.

Finally, while this model is inspired by the human mind and we draw illustrative analogies to it in this work, we make no claims that this is an accurate model of the human mind itself. We do posit that it is one lens through which to view a mind, whether natural or artificial, and that it could serve as a useful blueprint for designing computational minds.

Modeling Knowledge Acquisition via Sensory Grounding

In this section we present a formal model of a mind's knowledge base that is grounded in atomic sensory experiences of natural phenomena. Grouping, labeling, and categorizing sensory experiences gives rise to knowledge concepts, which can be further combined into higher-level concepts. Ultimately grounding these concepts in environmental phenomena provides concrete points of reference for manipulating and sharing concepts between minds, which is a chief concern of creativity.

Wiggins' formalization (Wiggins 2006) of Boden's model of creativity (Boden 2004) introduces a universe \mathcal{U} consisting of all possible concepts. Such concepts serve as the atomic unit of Wiggins' framework, and we expand upon his definition to model how concepts are represented in minds. Our concept model is thus compatible with Wiggins' model of conceptual systems as they pertain to exploratory and transformational creativity, while also expanding to incorporate expression and aesthetic evaluation which we will explore in a later section. We begin with an abstraction of natural phenomena and model how a mind can convert such phenomena into concepts.

Sensory Grounding

Let \mathcal{S} represent the set of all senses available to a given mind by which it may perceive its environment. For example, for the human mind this set could include the five basic senses commonly taught to children as well as the many more nuanced senses humans possess such as temperature, depth perception, and emotional sensations. Regardless of how the methods by which a mind experiences the universe are partitioned into senses, it necessary that \mathcal{S} include every one of those experience-enabling mechanisms.

Let a given sense $S \in \mathcal{S}$ represent a multi-dimensional *sense space* in which each distinct *sense point* $s \in S$ corresponds to a distinct instance of that sense. Again using human senses as an example, a sense point corresponds to a

particular smell, tone, hue, emotion, etc. Furthermore, this concept of sense spaces includes comparative differences in otherwise similar sensory information such as intensity. Gärdenfors presents a model of representing information in terms of quality dimensions based on sensory perception which could be viewed as one method for modeling sense points more granularly (2004).

Let I represent the set of all possible sense points regardless of which sense space they occur in, such that

$$I = \bigcup_{S \in \mathcal{S}} S.$$

Let the set \mathcal{P} represent all possible natural phenomena, which we define as quantized elements of the environment in which the mind exists. For example, in the real world a flower exists of itself and may be experienced in different way by different human senses such as sight, touch, and smell.

Such phenomena exist independently from the minds that observe them, and minds experience phenomena by way of their senses. Therefore, let

$$\psi : \mathcal{P}(\mathcal{P}) \rightarrow \mathcal{P}(I)$$

where \mathcal{P} is the power set, represent an observation function by which the phenomena that a mind encounters are experienced sensorily (i.e., translated into sense points). Thus, a given phenomenon $p \in \mathcal{P}$ is interpreted by ψ to yield a set of zero or more sense points, which may occur in disparate sense spaces.

This function ψ is immutable and mechanical, not consciously directed by the mind. Recognizing that senses are inherent and out of the mind's control is an important consideration for modeling and reasoning about minds. However, ψ may differ between distinct minds, even those of the same structure, resulting in different perceptions of identical phenomena. This corresponds, for example, to how a person with colorblindness perceives color differently than one without.

In a later section we will discuss phenomena that are generated by a mind. We note here that such phenomena necessarily exist in the same environment as "natural" phenomena, and as such we draw no distinction between them in this model.

Knowledge Acquisition

With a model for sensory experience of phenomena in hand, we turn to modeling the process by which a mind organizes sense points into concepts. Such concepts may be very complex and draw from many disparate sense points. We acknowledge the difficulty of building a robust model of knowledge acquisition and present this model as a means to abstract and compartmentalize the various mechanisms at play so that they may be addressed independently.

Let K represent the set of all concepts in a given mind's knowledge base, with $K \subseteq \mathcal{U}$, where \mathcal{U} is Wiggins' universe. We inductively define a concept $c \in K$ as follows:

Say that c is a **concept** if c is

1. $\{\} = \top$, the empty concept,

2. $\{s\}$ for some $s \in I$,
3. $c_1 \cup c_2$, where c_1 and c_2 are concepts, or
4. $\{c\}$, where c is a concept.

Note that a concept may simply consist of a set of sense points. Such low-level concepts directly describe natural phenomena (as filtered through the mind's senses), such as the concept of the color blue in the human mind. Higher-level concepts may contain a mix of sense points and other concepts, with the highest-level concepts being exclusively comprised of other high-level concepts.

We now turn to modeling the method by which a mind's knowledge base is assembled from the sensory input it experiences. Although minds may develop very different knowledge bases depending on the phenomena each mind experiences and the senses by which they perceive such, our model assumes that minds of the same structure acquire knowledge via the same underlying process.

Modeling this process is a daunting task due to the complexities of both the inputs into a mind and the network of knowledge that results from performing the knowledge-assembling process on those inputs. Our model abstracts those complex inputs as sense points which are used to assemble a knowledge base consisting of abstract concepts, facilitating the modeling of the the knowledge-assembling process itself.

We model the relationship between sense points and a mind's knowledge base via two functions: an *add* function, and a *lookup* function. The add function modifies the knowledge base to incorporate new sense points, creating new concepts or modifying existing concepts as necessary, and the lookup function determines what concepts a set of sense points evokes in the mind, respectively. Both of these functions are nontrivial to model and compute, but we present an abstraction for the latter.

Let $\lambda : \mathcal{P}(I) \rightarrow \mathcal{P}(K)$ represent the lookup function by which a mind relates sense points to existing concepts in its knowledge base. Given a set of sense points $Y \subseteq I$, the set of concepts returned by $\lambda(Y)$ may be very simple (i.e., a subset of I) or may consist of potentially many combined high-level concepts. Modeling the means by which a knowledge base is traversed from low-level concept representations of sense points to high-level concepts that represent a synthesis of those points appears to be a challenging avenue for future work.

Our abstraction of the "add" function does not lend itself to further subdivision as it directly mutates the knowledge base to incorporate new sense points. As such, we leave exploration of this function as future work. Describing the process by which a knowledge base is constructed and augmented as the mind encounters new sense points is central to implementing this knowledge model. As stated previously, our intent in presenting this model is to compartmentalize and isolate difficult aspects of modeling minds to reduce their complexity as much as possible.

Expression, Perception, & Aesthetics

A mind does not acquire knowledge simply to hoard it; its knowledge base is a deep well of resources that informs and

powers other useful mental faculties. One of the most direct utilizations of knowledge is the act of expression, which we define as the production of phenomena meant to evoke certain concepts.

Creativity, which is the primary focus of this work, is encompassed within the larger umbrella of expression. Although creatively combining concepts or artfully expressing concepts in novel ways may differ substantially from merely reciting simple or well-known concepts, we draw no distinction between creative and non-creative expression in this model. Both can be modeled as the conversion of knowledge concepts into phenomena.

Expression

Expression can be modeled as a function that maps concepts to phenomena. Let $\xi : \mathcal{P}(K) \rightarrow \mathcal{P}(\mathcal{P})$ represent an expression function by which a mind generates phenomena from concepts. For example, the human mind has various modes of expression that may result in different phenomena representing the same concepts, such as drawing a picture of a bird versus flapping one's arms to imitate a bird in flight.

Let Ξ represent the set of all expression functions available to a given mind. This model encapsulates all differences in type of expression between the various functions whether they be in the type of phenomena they produce (i.e., the medium) or the quality of those phenomena. For example, two humans' functions for expressing concepts via paintings could differ due to their different experience levels with working in that medium.

Expressing concepts as phenomena is a challenging task to which there exist a large number of potential approaches and solutions. The human mind accomplishes expression via a physiological linkage between the brain and other body parts and requires practice to attain proficiency. Identifying useful ways to computationally generate phenomena from concepts is an ongoing quest in computational creativity and other artificial intelligence disciplines, and in fact the invention of novel expression functions could itself be an interesting creative task. By abstracting the other complexities of a mind's knowledge base, our model seeks to isolate and, to the extent possible, simplify this act of expression for further investigation.

Perception

The way other minds perceive phenomena generated through expression is an important consideration for expressive and creative endeavors. In particular, it is useful to compare the set of concepts $C \subseteq K$ from which the expresser generates phenomena and the set of concepts that those phenomena elicit in a perceiver.

Given two minds a and b , a set of concepts C_a that a wishes to express, a 's expression function ξ_a , and b 's observation and lookup functions ψ_b and λ_b , let

$$C_b = \lambda_b(\psi_b(\xi_a(C_a))).$$

Thus, C_b is the set of concepts that b infers from the phenomena a generates when attempting to express C_a via ξ_a .

By comparing C_a to C_b , we can model how closely b understood the concepts that a intended to express. We hypothesize that λ lookup functions will commonly be more successful at linking phenomena to the intended concept when a lower-level concept is being expressed. Higher-level concepts require the traversal of more connections from sense points to final concept and therefore seem more likely to be misunderstood.

In the special case that $a = b$, this process results in

$$C'_a = \lambda_a(\psi_a(\xi_a(C_a)))$$

which represents a mind evaluating its own expression, perhaps to compare how well the phenomena it generated reflect its intended concepts. This models a common and useful operation in creativity processes.

Aesthetics

The creative process often includes elements of aesthetic evaluation to complement generation. Such evaluation can be applied to one's own creative works as well as to those of others. Indeed, aesthetic appreciation is a significant factor in what gives meaning to some important creative endeavors.

We model the aesthetic sensibilities of a mind as partitions of its sense spaces, with each partition being considered of different aesthetic quality by the mind. For a simplified example, consider a human mind that finds certain smells appealing and others off-putting. This corresponds in our model to a partition of that mind's olfactory sense space into two subsets.

Let Θ represent a mind's set of aesthetic partitions of its sense spaces, such that

$$\forall \theta \in \Theta. \exists S \in \mathcal{S}. \theta \subseteq S$$

and

$$\forall S \in \mathcal{S}. \exists T \subseteq \Theta. \bigcup_{\theta \in T} \theta = S.$$

Note that this model requires complete coverage of each sense space by aesthetic partitions. Thus, a mind will always have an aesthetic opinion of any newly encountered sense point. Descriptions of the extent to which a mind is aware of its aesthetic partitions, the degree to which those partitions may change over time, and the degree to which such changes can be consciously enacted by the mind are left to future work.

We assume that the mind maintains an aesthetic opinion for each θ , perhaps by means of a function $q : \theta \rightarrow \mathbb{R}$ that maps a partition to an abstract, real-number representation of the mind's opinion of the sense points in that subset. We concede that aesthetic opinion may be more nuanced than can be represented by a single real number and leave further exploration into useful models of such opinion as future work.

Let $\pi : I \rightarrow \Theta$ represent a function that identifies to which aesthetic partition a sense point belongs. Then, to model a mind's aesthetic opinion of a set of phenomena $P \subseteq \mathcal{P}$, construct

$$A = \{q(\pi(s)) \mid s \in \psi(P)\}$$

which contains the set of aesthetic opinions that correspond to the sense points experienced from those phenomena. Constructing A for a set of phenomena generated by an expresser's ξ reflects the perceiver's aesthetic assessment of that expression.

Similarly to calculating C'_a to evaluate the concepts evoked by a mind's own expressed phenomena, constructing A for those same phenomena allows the mind to aesthetically evaluate that expression. This "self-criticism" operation is useful in creative processes.

Our model also allows for the aesthetic assessment of (grounded) concepts. Let $c \rightsquigarrow s$ represent the *grounded in* relation between a concept c and a sense point $s \in I$. Say $c \rightsquigarrow s$ if and only if

1. $s \in c$ for some $s \in I$ or
2. $b \in c$ and $b \rightsquigarrow s$

Then, given a concept c , constructing O such that

$$O = \{q(\pi(s)) \mid c \rightsquigarrow s\}$$

allows us to model the mind's aesthetic opinion of the concept in question via the sense points that ground the concept.

Discussion

Our model of knowledge is by no means exhaustive. A mind's knowledge base influences many other important mental functions such as language, reasoning, and imagination.

Language in particular is interesting to consider under our model. Sensory experiences of language phenomena seem to circumvent a mind's add and lookup functions by serving as "machine code" that executes on the knowledge base directly. We view language and other facets of minds as interesting avenues for future work to be explored using our knowledge model.

This model seeks to be environment- and mind-agnostic so that it allows for many interpretations. As such, it may be useful to explicitly model a given mind's *environment* as $E \subseteq \mathcal{P}$ in order to reason about the types of phenomena that that mind can and cannot observe.

If minds a and b exist in environments E_a and E_b , respectively, such that $E_a \neq E_b$, then there exist some phenomena which each mind can experience that the other cannot. As phenomena inform sense points which in turn inform concepts, this means that there could exist concepts that the minds cannot share. Identifying the differences between human and computational environments could give better insight into the limitations of direct human-to-computer communication, and vice versa.

We believe that further exploration of this model of knowledge that is grounded in sensory perception of environmental phenomena will be useful for designing computationally creative systems that must ultimately operate with in the environment of the natural world as perceived by human senses.

References

Boden, M. A. 2004. *The creative mind: Myths and mechanisms*. Routledge.

- Colton, S., and Wiggins, G. A. 2012. Computational creativity: The final frontier? In *Proceedings of the European Conference on Artificial Intelligence*, 21–26.
- Gärdenfors, P. 2004. Conceptual spaces as a framework for knowledge representation. *Mind and Matter* 2(2):9–27.
- Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv abs/1301.3781*.
- Miller, G. A. 1995. WordNet: A lexical database for English. *Communications of the Association for Computing Machinery* 38(11):39–41.
- Speer, R., and Havasi, C. 2012. Representing general relational knowledge in ConceptNet 5. In *Proceedings of the International Conference on Language Resources and Evaluation*, 3679–3686.
- Wiggins, G. A. 2006. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems* 19(7):449–458.

Exploring the Notion of Self in Creative Self-Expression

Vikram Jamwal

TCS Research

Pune, INDIA

vikram.jamwal@tcs.com

Abstract

Creative self-expression is considered as central to any artistic activity. It is governed by an inherent need in a person to express *the self* through a creative or an artistic medium, e.g., visual art, poetry, music, drama, design, etc. In this paper, we explore and provide a classification of the *notion of self* from the various perspectives of philosophy, psychology, modern science, etc. This is done with a view to understand the self holistically and to frame a definition in the context of ‘creative self-expression’. We also aim to initiate a discussion on the relevance of the concept of self to the field of computational creativity.

Introduction

Perhaps best known for his painting ‘*The Scream*’, Edvard Munch (1863 to 1944) is considered by many as the most iconic figures in the modern art world after Leonardo Da Vinci (Friedlaender and Friedlaender 2018). Many of his works explore intense psychological themes. He suffered from deep depression during his lifetime and his art often reflected events that happened to him. Post his difficult childhood, Munch started channelising his inner turmoil to create art. He was clear about his mission; it was to explore the portrayal of extreme human emotion: “*Just as Leonardo da Vinci studied anatomy and dissected corpses so I try to dissect souls*”.

Explaining the context in which ‘*The Scream*’ was conceived, Munch wrote in his journal ¹: “*I was walking along the road with two friends. The sun was setting. I felt a wave of sadness. The sky suddenly turned blood-red. I stopped, leaned against the fence tired to death, gazed over the flaming clouds like blood and swords, the blue-black fjord and city. My friends walked on. I stood there trembling with angst, and I felt as though a vast, endless scream passed through nature.*” Commenting on Munch’s writings, historian J Gill Holland remarks: “In his journal entries he was on the prowl for the unmediated transmission of mind to page” (Munch 2005).

We can say that Munch’s art was a reflection of the world that he experienced and a reflection of his self. In his art, he

¹MM T 2760, Munch Museum. Date 1891-1892. Sketchbook (2019-05-31)

endeavoured to bring his self to the fore. He stated: ‘*I do not believe in the art which is not the compulsive result of Man’s urge to open his heart.*’ Considering all this, we are led to believe that Edvard Munch’s work can be considered as an epitome of what we may call as *creative self-expression*.

Creative Self-Expression(CSE)

Self-expression is a pervasive phenomenon of everyday life of humans and many other species. Self-expression reveals our states of thought, feeling, and experience to others (Green and others 2007). It is fundamental to human society; it helps us understand, empathise and communicate with other fellow beings. It is achieved through facial expression, language, body language, speech acts, etc. *Creative self-expression* puts the additional requirement of ‘creativity’ in self-expression. It means an expression of one’s self - whether it be its ideas, feelings, or personality - creatively or through creative art forms. The form of expression would differ with the domain or the medium of expression - visual art, music, poetry, painting or anything similar.

In computational creativity (CC) research, we have so far emphasised the ‘*creative*’ part of the CSE concentrating mostly upon the generation of creative artefacts, the techniques of producing creative artefacts, or, more recently, the ‘intentions’ of producing creative artefacts. However, we feel that we have not paid adequate attention to the core part of CSE, viz., the ‘self’. The two most important questions that relate to self in this context are: (1) the *Who* question: Who is that ‘self’ that wants to express? (2) the *What* question: What is ‘of that self’ that needs to be expressed?

This paper motivates a discussion of that gap. As is evident from Munch’s example and the lives of many others: For an artist, *self-expression* forms the essence of a creative act. We believe that besides ‘creativity’, ‘self-expression’ should form a part of the core investigation in computational creativity research. Hence, the notion of ‘self’ should be recognised and designed as a first-class entity in a computational creativity system - particularly if that system also professes to be a self-expressing agent. In this paper, we dwell upon the ‘notion of self’ and draw upon the various views of self from the perspectives of philosophy, psychology, modern neuroscience, etc. We further discuss this in the context of research in Computational Creativity.

Notions of Self

When you play music you discover a part of your-self you never knew existed. - Bill Evans

The question, 'What exactly is self?', elicits many diverse answers. It has been the subject of philosophy, psychology, science and religious discourse over the human history. There is no one accepted answer (Olson 1999). Still, we have to find a definition that is universally understandable (if not universally accepted), and which can form the basis of our movement in the field of computational creativity. In this quest, we present a classification drawing upon various viewpoints from philosophy, psychology, neuroscience, etc.

Philosophy Views of Self This section summarises the prominent views on the nature of self postulated by various philosophers over the last few centuries.

Solitary Self / Dualism View: In modern philosophy, *Rene Descartes* is believed to have been the first one to establish the idea of solitary self. He introduced the idea of dualism stating that human beings consist of mind (soul) and body and it is the 'thinking' that becomes the defining characteristic of our self or our existence; 'Cogito, ergo sum', or, 'I think, therefore I am' (Descartes 1641).

Memory as Self-identity View: Philosopher *John Locke* differs from Descartes where he distinguishes between the substance (soul) and consciousness (Locke 1860). According to him, self-identity results from having the same memory (or consciousness). It is the memory that provides the definite link binding together different stages of a person. If we perceive that we are the same person from one time to another, it is not as a result of us having the same body or the same substance, but as a result of us having the same consciousness. Two often cited objections to Locke's view are: (1) our memories are not correct or precise all the time, and (2) we forget a large part of our conscious experience.

Bundle-Theory View: An alternative version of the self is based on the 'bundle theory' by the Scottish Enlightenment philosopher, *David Hume*. Hume believed that the idea of self is a fiction. Humans do not have an actual conception of self. There is only a bundle of sensations, perceptions and thoughts which are piled on top of each other. The self emerges only out of bundling together of these experiences (Hume 1739).

Transcendental Self and Empirical Ego View: *Immanuel Kant* is in agreement with Hume on that self-identity does not come from self-consciousness. Kant, however, opines that the 'enduring self' is transcendental and not just an object of experience (Kant 1781). By transcendental Kant implies the necessary condition for the possibility of any experience. He argues that if there is a separate self at each moment of experience, we will not be in a position to perceive anything. There has to be a unified consciousness that combines all these perceptions; that, according to him, is self. Unlike Descartes, Kant believes that the self is not an experience but rather it is one that is responsible for the experience. So he proposes two different conceptions of self: (1) *Empirical ego* - includes all those specific things that make us different people, and (2) *Transcendental self* - that which is essential to a unified empirical self-consciousness.

Ego-theory / Pearl View: This is the view that an average person on the street would most likely have about her self: Self is the individual that inhabits the body and body is something that is controlled by the self. Self is the essential entity at the core of our existence that holds steady throughout our life. The ego experiences life as a conscious, thinking person with a unique historical background and defines who we are. The philosopher Galen Strawson describes it as 'Pearl view' of the self suggesting that many mental selves exist, one at a time and one after another, like 'pearls strung on a thread' (Strawson 1997).

Psychology Views of Self One of the earliest formulations brought out by the modern psychology is the distinction between two aspects of self: self as a subjective knower - *Self as I* - the one which is conscious and aware of experience, and self as the object that is known - *Self as Me* - the one which is understood as 'personal identity'. Here we present some predominant views in psychology on self.

Comprehensive Self View: *William James* (James 1890) puts forth that a man's self is a sum of all that he can call his: (1) *Material self* - his body, clothes, family, lands, possessions, work of his hands; (2) *Social self* - the recognition which he gets from his mates - "a man has as many social selves as there are individuals who recognise him and carry an image of him in their mind"; (3) *Spiritual self* - man's inner or subjective being, his psychic faculties or dispositions; includes the faculties and the entire stream of personal consciousness. It results from the reflective process of abandoning the outward-looking point of view; (4) *Pure ego* - the bare principle of personal unity. In short, according to James, our sense of self extends from the *immediacy of our experience to the contemplation of innermost thoughts*.

Looking-glass Self View: Building up from the work of *W. James*, *Charles H. Cooley* surmised that there is an indissoluble link between self and society. Our self does not exist independently but as a reflection of those around us. Outside perceptions have an impact on who we think we are. We not only learn from others but we also learn to become like others. This idea is sometimes called the looking-glass self (Cooley 1902). These influences work right from childhood. Even as adults we continually develop and elaborate our internal definition of self. This definition might also be multifaceted depending upon the roles that we take and the external world contexts that we handle - the work self, the home self, the parent self, the political self, the bigoted self, the emotional self, the sexual self, the creative self, the violent self, etc. (Hood 2012).

Self-Knowledge View: Another popular conception of self comes from *Ulric G. Neisser* (Neisser 1997). He formulates that there are five distinct levels of self-awareness and each of these establishes, in essence, a different self, viz., (1) *Ecological self* - based on perceptual cues - visual, auditory, and kinaesthetic; (2) *Interpersonal self* - based on social interactions; (3) *Extended self* - based on memory and anticipation; (4) *Private self* - based on processing of thoughts, feelings, intentions, etc. as an exclusive personal experience; (5) *self-concept* - based on abstract and symbolic representation of oneself like role, traits, identity, etc.

Contemporary Brain Science Views of Self Most neuroscientists reject the idea that self exists independently of body and brain. If the self is the sum of our thoughts and actions, then it is also true that these depend on the brain. Thoughts and actions are not exclusively of the brain; we also think about and act upon things in the world with our bodies. However, the brain is primarily responsible for coordinating these activities. These following views then take credence:

Self as an Illusion: There is no centre in the brain where the self is constructed. The brain has many distributed jobs: (1) it processes incoming information from the external world into meaningful patterns that are interpreted and stored for future reference; (2) it generates different levels and types of motivations, i.e., the human drives, emotions and feelings; (3) it produces all sorts of behaviours - some of them automatic while others are acquired through skill, practice and sheer effort.

The sense of self that most of us experience is not to be found in any one area. Rather it emerges out of the orchestra of different brain processes like a symphony of the self (Hood 2012): “Our brain creates the experience of our self as a model - a cohesive, integrated character - to make sense of the multitude of experiences that assault our senses throughout a lifetime and leave lasting impressions in our memory”. Our brain constructs models of the external world. It weaves experiences into a coherent story that enables us to interpret and predict our next recommended action. In short, our brain simulates the world to survive in it. Hence, modern neuroscience tends to support the bundle theory more as opposed to the ego theory of the self.

Self as a Centre of Narrative Gravity: Cognitive scientist *Daniel Dennett* thinks that who we are is a story of our self. This self is constructed out of narratives of our brain: “Our tales are spun, but for the most part, we don’t spin them; they spin us. There is no self at the core. Rather it emerges as the *centre of narrative gravity*” (Dennett 2014).

However, a hard problem that remains unsolved presently is that we do not know how a physical system like the brain can produce a non-physical experience like the conscious self. Some philosophers believe that an answer to this question might be elusive or the question might itself be misguided (Chalmers 1995).

Self in Self-Expression

In the previous sections, we presented the notion of self from various perspectives - philosophy, psychology, neuroscience, etc. As would be evident from the above discussion, these views are not always compatible. Nor are they mutually exclusive. Our purpose of presenting these various points of view was to give us a broader understanding of self. We would like to judiciously reflect upon the question: Which of these views (and under what circumstances) are most relevant to *creative self-expression (CSE)* in general, and *machine-driven CSE* in particular? While it is difficult to put out one single definition of self, we propose the following framing for self which we believe can harmonise these various views. For a person, involved in creative self-expression, the self can be considered to consist of:

(1) *The Actual Self:* What a person ‘actually is’ with respect to the person’s body, brain and mind, or even transcendence (assuming such a possibility can be accounted for).

(2) *The Acquired Self:* The experiences, learnings, reflections, action-reactions, and imaginations, etc., that a person would have had through the course of life.

(3) *The Personal Notion of Self:* The very notion or definition of self embedded in the person’s mind.

The third one is a *meta notion* - the notion that a self has about itself. The philosophical notions we carry about ‘our self’ may have a strong impact on our creative outputs. For instance, some early British women novelists (like Aphra Behn, Jane Barker, Eliza Haywood, and Mary Davys) were influenced by the debates about ‘self’ generated during the ‘scientific revolution’; and this shaped the narrative practices within the early novels of that period (Gevirtz 2014).

Notion of ‘Inner Point of View’

“I shut my eyes in order to see.” - Paul Gauguin

Here, we further explore and elaborate on the objective aspect of self with the question: “What exactly is expressed in a creative self-expression?” As humans we think, we feel, and we experience. The perceptions of the outer world are processed by (subjective) self’s faculties of aesthetics, imagination, introspection, reflection, etc. to create an inner world. This is understood as the ‘Me’ of the objective self - my thoughts, my feelings, my experiences. This created inner world, which we may refer to as the *‘inner point of view’*, is private to the individual. It remains so until and unless she chooses to share it with the rest of the world. There is no direct sharing though - I cannot think your thoughts, I cannot feel your feelings, I cannot experience your experience. But it can be made available to others as an ‘expression’. The receiver relates to it through the faculty of empathy - by interpreting it in the light of her own inner truths.

Each self has a unique moulding - individual differences in temperament, personality, intelligence, perception, personal background, and experiences. This accounts for the wide range of variations we see in artistic capacities, expressions and tastes (Shimamura and Palmer 2012). ‘Self-expression’ thus is presenting a part of the content of your unique self - personality, thoughts, feelings, experiences, opinions, stories, etc. - to the outer world. It is this inner point of view that constitutes: “What is that that is to be expressed?” Note that ‘what is to be expressed’ might be vague in the beginning and a clearer view of the content might unfold only during the process of expression (Hospers 1954).

Articulation of the inner point of view is always through a medium. The modality of expression, i.e., what can be effectively conveyed and how it can be conveyed, will differ from medium to medium. For example, in case of representational arts (such as painting, movies, and literature, etc.) one needs to understand what they are about (i.e., what the painting or movie depicts, and what the poem or the novel describes) (Robinson and Hatten 2012); while in the case of music, which lacks semantic or representational content (Davies 2006), people value it primarily because of the emotions that it evokes (Juslin 2013).

Discussion on Related CC Concepts and Work

Computational systems, including CC systems, presently do not explicitly support the notion of ‘self’ as underlined in this paper. Though the term ‘self-expression’ appears in Self-aware systems, it is used in a limited sense; it is used to describe self-adaptive behaviour that is based on the knowledge acquired through system self-awareness (Torresen, Plessl, and Yao 2015). Below, we consider some important and related points-of-view in the CC literature and discuss their relevance to the self in creative self-expression.

Creativity Tripod: Earlier efforts in CC were concerned more with the generation of a creative artefact, without the creative agent (e.g. a poem generator) having an appreciation of what it was doing or what it has produced (e.g. the semantics of the poem). (Colton 2008) argue that this amounts to having no creativity at all. The authors in (Charnley, Pease, and Colton 2012) hypothesise that a *creativity tripod* of skilful, appreciative and imaginative behaviours are the bare minimum required to support the perception of creativity in computational systems. It is similar to how one would assess the creativity in a human (e.g., a poet, a painter). We agree with the above, but feel that these are not sufficient conditions for CSE. Instead of ‘perception of creativity’, CSE puts a primary focus on the ‘expression of self’. Focus on ‘self’ implies that the agent is a live entity in an environment, builds an inner point of view and is capable of expressiveness.

Notion of Framing: The way artists explain their work - in terms of *motivation* (why did you do X?), *intention* (what did you mean when you did X?), and *processes* (how did you do X?) - has a very large influence on how the audience perceives them. Framing captures this information (Charnley, Pease, and Colton 2012). For example, Munch’s diary notes on ‘The Scream’ give further insight to the viewers on how to interpret his works. A work of art might even have a completely different interpretation if we change parts of the framing information. We feel that for a well-implemented CSE, framing information would be easily available and a natural consequence. This is because the agent, who is also self-aware (a requirement in our case), has to only speak out its truths and should be able to reflect upon and answer the questions on motivations, intentions and processes.

FACE Model: FACE descriptive model, put forth in (Colton, Charnley, and Pease 2011) and (Pease and Colton 2011), advocates describing a creative system in terms of the creative acts (tuples of generative acts) it performs. The generative acts produce four types of outputs: examples of concepts, concepts themselves, aesthetic measures which can evaluate concept/example pairs, and framing information. We believe that the FACE model, as applied to CC systems, should be easily amenable to extend to CSE systems. As mentioned earlier, framing information is a natural consequence in a CSE system. FACE model already carries the notion of ‘concept’ and its conversion to ‘expression’. In case of CSE, the emphasis would be on converting something that is a part of ‘the self’ (*idea, feeling, experience, etc.*) to something tangible in the external world (*musical piece, visual art, poem, design, etc.*). Extension of FACE model, as to be relevant to CSE, is a subject of future study.

4P Perspective in CC: The authors in (Jordanous 2016) argue that to consider creativity holistically - consideration of mere process and product is not enough; computational creativity research (as is creativity research (Rhodes 1961) (MacKinnon 1970)) should be considered and explored from four different perspectives, known as the *Four Ps: Person, Product, Process, Press*. Articulation of ‘person’ in (Jordanous 2016) is more of a ‘producer’ (authors propose in this paper: ‘the term Producer is more appropriate as it allows us to consider the Four Ps in the contexts of both human and computational creativity’). In our view, for CSE, the self is more than just a producer of the artefact - it is an experimenter of the world, an introspective reflector, imaginer, as well as a creative producer.

CC Continuum: The paper (y Pérez 2018) argues that the construction of creative systems is motivated by, what sometimes seems to be, diverse and even contradictory viewpoints and understandings about the goals of computational creativity. One pole is the *engineering-mathematical* approach and the opposite pole is the *cognitive-social* approach. Any creative agent is located along the *continuum* based on its main goals as a system. In our view, an agent capable of CSE and supporting the necessary notion of ‘self’ would more likely be built upon cognitive-social approaches and hence would fall onto this side of the spectrum.

Applications and Systems: A recent study (Loughran and O’Neill 2017) reviews the diverse range of applications (and systems) considered in CC. Some application systems, e.g. Painting Fool (Colton 2012) give a semblance of a quasi sense of self. For instance, it has demonstrated the capability to get into certain moods during its painting efforts. A model for meta-creativity based on self-awareness is presented in (Linkola et al. 2017). Here self-awareness is taken as the capacity to become the object of one’s attention with a potential to also change oneself. However, to our knowledge, presently no direct conceptualisation or implementation of ‘self’ for ‘creative self-expression’ has been formally considered or incorporated in any of the CC systems.

Conclusion

In this paper, we discussed how expressing *our self* creatively has been one of the principal motivations for any artistic endeavour in human history. We explored the *notion of self* from the various viewpoints of philosophy, psychology, and modern science and presented a classification of the same. We pointed out how these concepts are relevant to *creative self-expression (CSE)* and how CSE helps to explore, articulate, and even enhance the self. We strongly feel that the notion of *self* in *self-expression* should form the subject of core investigation in computational creativity research and hope that this paper initiates that discussion. We further believe that the systems supporting these notions would lead to better human-machine artistic and creative collaborations and a greater value being assigned to the machine created artefacts. A survey of the existing CC systems with a view on their capability for creative self-expression, and exploring the possible ways to incorporate the relevant notion(s) of self in CC systems are areas of future work.

References

- Chalmers, D. J. 1995. Facing up to the problem of consciousness. *Journal of consciousness studies* 2(3):200–219.
- Charnley, J.; Pease, A.; and Colton, S. 2012. On the notion of framing in computational creativity. In *International Conference on Computational Creativity*, 77. Citeseer.
- Colton, S.; Charnley, J. W.; and Pease, A. 2011. Computational creativity theory: The face and idea descriptive models. In *ICCC*, 90–95.
- Colton, S. 2008. Creativity versus the perception of creativity in computational systems. In *AAAI spring symposium: creative intelligent systems*, volume 8.
- Colton, S. 2012. The painting fool: Stories from building an automated painter. In *Computers and creativity*. Springer. 3–38.
- Cooley, C. H. 1902. Looking-glass self. *The production of reality: Essays and readings on social interaction* 6.
- Davies, S. 2006. Artistic expression and the hard case of pure music. *Contemporary Debates in Aesthetics and the Philosophy of Arts*.
- Dennett, D. C. 2014. The self as the center of narrative gravity. In *Self and consciousness*. Psychology Press. 111–123.
- Descartes, R. 1641. Meditations on first philosophy. *Meditationes de prima philosophia, in qua Dei existentia et animae immortalitas demonstratur*.
- Friedlaender, G. E., and Friedlaender, L. K. 2018. Edvard munch and the scream: A cry for help. *Clinical orthopaedics and related research* 476(2):200.
- Gevirtz, K. 2014. *Women, the novel, and natural philosophy, 1660–1727*. Springer.
- Green, M. S., et al. 2007. *Self-expression*. Oxford University Press.
- Hood, B. 2012. *The self illusion: How the social brain creates identity*. Oxford University Press.
- Hospers, J. 1954. The concept of artistic expression. In *Proceedings of the Aristotelian Society*, volume 55, 313–344. JSTOR.
- Hume, D. 1739. 1739/1888. *A Treatise of Human Nature*.
- James, W. 1890. Principles of psychology (vol. 1). new york, ny: Henry holt and company.
- Jordanous, A. 2016. Four PPP perspectives on computational creativity in theory and in practice. *Connection Science* 28(2):194–216.
- Juslin, P. N. 2013. What does music express? basic emotions and beyond. *Frontiers in psychology* 4:596.
- Kant, I. 1781. Critique of pure reason. *Modern Classical Philosophers, Cambridge, MA: Houghton Mifflin (1908)* 370–456.
- Linkola, S.; Kantosalo, A.; Männistö, T.; Toivonen, H.; et al. 2017. Aspects of self-awareness: An anatomy of metacreative systems. In *Proceedings of the 8th International Conference on Computational Creativity (ICCC'17)*. Georgia Institute of Technology.
- Locke, J. 1860. *An essay concerning human understanding: and a treatise on the conduct of the understanding*. Hayes & Zell.
- Loughran, R., and O'Neill, M. 2017. Application domains considered in computational creativity. In *Proceedings of the 8th International Conference on Computational Creativity, Atlanta*.
- MacKinnon, D. W. 1970. Creativity: A multi-faceted phenomenon. *Creativity* 19–32.
- Munch, E. 2005. *The private journals of Edvard Munch: we are flames which pour out of the earth*. Terrace Books.
- Neisser, U. 1997. The roots of self-knowledge: Perceiving self, it, and thou. *Annals of the New York Academy of Sciences* 818(1):19–33.
- Olson, E. T. 1999. There is no problem of the self. *Models of the self* 49–61.
- Pease, A., and Colton, S. 2011. Computational creativity theory: Inspirations behind the face and the idea models. In *ICCC*, 72–77. Citeseer.
- Rhodes, M. 1961. An analysis of creativity. *The Phi Delta Kappan* 42(7):305–310.
- Robinson, J., and Hatten, R. S. 2012. Emotions in music. *Music Theory Spectrum* 34(2):71–106.
- Shimamura, A. P., and Palmer, S. E. 2012. *Aesthetic science: Connecting minds, brains, and experience*. OUP USA.
- Strawson, G. 1997. The self. *Journal of Consciousness Studies* 4(5-6):405–428.
- Torresen, J.; Plessl, C.; and Yao, X. 2015. Self-aware and self-expressive systems. *Computer* 48(7):18–20.
- y Pérez, R. P. 2018. The computational creativity continuum. In *Proceedings of the Ninth International Conference on Computational Creativity ICCCI*, 177–184.

Region Radio: An AI that Finds and Tells Stories about Places

Douglas H. Fisher Emily Markert Abigail Roberts Kamala Varma

Department of Electrical Engineering & Computer Science
Vanderbilt University
Nashville, TN 37235
douglas.h.fisher@vanderbilt.edu

Abstract

Region Radio is an artificially intelligent platform that finds and tells stories about the places that a user is moving through -- by car, bicycle, foot, or any mode of transport. Stories will typically be non-fiction with environmental and historic-cultural themes that are intended to increase a conservationist consciousness on the part of listeners and content contributors. Region Radio finds its stories on the Web, based on their relevance to places that lie on a route specified by the user; filters and ranks these stories on projected relevance; schedules these stories onto a playlist that corresponds to the route; and reads these stories as the user travels the route (to include “travel” in the minds-eye).

Introduction

Region “Radio” is an artificially intelligent platform that finds and tells stories, typically non-fiction, to people about the places that they are moving through -- by car, bicycle, foot, or any mode of transport. Region Radio finds its stories on the Web, based on relevance to locations that lie along a route specified by the user; filters and ranks these stories for projected relevance; schedules these stories onto a playlist of stories that corresponds to the route; and reads these stories as the user travels the route (to include “travel” in the minds-eye). “Place” is broadly construed to include the physical location, but also the people and events of the locale across time. Our current focus is conservationism, both environmental and historic-cultural, with the goal of increasing conservationist consciousness on the part of listeners and content contributors (Milligan, 2011).

We have developed an initial implementation of Region Radio, which as yet has not been thoroughly evaluated or deployed. The implementation is internally complete in terms of the most important basic functionality, though longer-term desiderata are not yet implemented.

Curation as Creation

The creations that Region Radio produces are playlists of existing stories that are found on the Web. The creative act is one of curating.

Outwardly, the playlist is a total ordering on selected stories, but should a user swipe past a story (not implemented)

associated with a place, another is selected from a location-sensitive priority queue of alternatives (implemented), and so the playlist is actually a partial ordering on stories. A playlist is a narrative, but one that is composed of loosely-coupled components (i.e., stories) as with most any curated collection. As narrative, we can assess the playlist in terms of characteristics like tension, polarity, emotion, and ultimately interestingness and user satisfaction. The latter characteristics have much to do with the characteristics of the individual stories, but also how the collection hangs together, which we discuss in Ongoing and Future Work. From a theoretical perspective, the **kind and amount of coupling** between components in a playlist are, perhaps, one extreme in a useful framework for studying creative narrative artifacts, where book chapters represent much greater coupling and reside in a different part of the proposed coupling spectrum/framework.

Region Radio has important connections to

- ♦ place-based education (Gruenewald & Smith, 2008);
- ♦ locative narrative (Greenspan, 2011);
- ♦ tour “guides” (Nisi, et al, 2008);
- ♦ points-of-interest notifications (e.g., Google Fieldtrip);
- ♦ configuration tasks (e.g., Maher, et al, 2016); and
- ♦ interactive story telling (Nisi, 2017), particularly in future designs.

Region Radio appears novel relative to other related efforts (in one or a combination of ways) in its *use of AI* to find place-relevant *stories*, with ambitions that these be *interesting* stories, from a *virtually limitless space*, and to *schedule these stories* into a playlist, with ambitions that the *interestingness of the playlist itself be greater than the sum of its parts* (i.e., the individual stories).

We believe that Region Radio will benefit three audiences.

Listeners

Region Radio is intended to educate users about places -- the geography, the environment, the people, the history, and the culture. The listener may be on a drive from Seattle, WA to Charlotte, NC. Alternatively, they may be on a frequently repeated walk/run in their community and its environs. Region Radio has enough intelligence to avoid telling the listener the same stories over and over, and

more generally, will customize a “podcast” to the user (but not yet) and the places in the regions that the user moves.

Authors

A distinct, and we hope overlapping population, are content developers -- “story tellers”. The stories that Region Radio finds on the Web can be from established sources (e.g., Nature Conservancy Magazine, community newspapers), but we view Region Radio as potentially incentivizing story authoring by community members of all types, including students, parents, grandparents, teachers, and churches. It is in this population of authors that we see the potentially transformative opportunities of Region Radio in place-based education. As an incentive mechanism, Region Radio adds to work on *spatial-digital story lines* or DSSLs (Hall, et. al., in press). Story authoring is a pipeline that includes collecting facts, memories, news, and insights, much like that in the map-based DSSLs. DSSL developers, and/or others, can translate these spatial narratives into text-based stories that can be found and told by Region Radio to the public on a potentially large scale. In the case of student authors of final textual stories, Region Radio can deliver their work to “authentic audiences” (Light, 2004), which may further incentivize them to produce accurate, comprehensive, and engaging material.

Other Curators

When Region Radio draws from, and acknowledges (as it always will), sources like magazines and newsletters, it supports these other curators and the cultural heritage that those curators help to maintain. As Madison (2011) notes “*analyses of creativity and innovation usually focus on producing new knowledge and offering access to it. Equivalent questions concerning existing knowledge, preserving and conserving old things and offering access to them, get less frequent attention.*” Because Region Radio draws from many other curators, it is intended to make listeners aware of these other curators, and to facilitate contributions to those other sources (see Ongoing & Future Work).

We turn to a system description of Region Radio, followed by an example playlist, ending with a discussion on basic research issues, and planned extensions.

System Description

Our initial implementation of Region Radio has been developed over 4+ months, distributed over the past two summers, by undergraduate researchers and developers. The implementation is in Python on a Github repository.

Figure 1 gives a high-level architectural diagram of Region Radio as it is currently implemented. Given a route to be travelled, the system constructs a playlist of stories about various places along the route. Each story in the playlist has an expected duration, and travel times are estimated as well. The playlist is constructed so that a story’s telling will be timed to correspond to a time when the cor-

responding places are being approached. The four depth-2 processes are arranged as nested loops, as suggested by the nested boxes, so that finding a suitable location from Location Analysis is informed by the stories that are associated with places in the vicinity of that location (i.e., through retrieval, evaluation, scheduling).

In this initial implementation, the story playlist is created offline, before actual travel, for upload to an appropriate listening device. We have represented the playlist as a partially ordered plan to facilitate dynamic adjustments in response to user decisions during travel, but at this writing we have not implemented a dynamic user interface that can exploit the partial-ordering of stories through explicit user selection of stories or through dynamic GPS updates on actual position. The travel route is created by Google Maps Directions API. In short, we describe the offline construction of a static playlist of stories, given a travel route that is an input to the system.

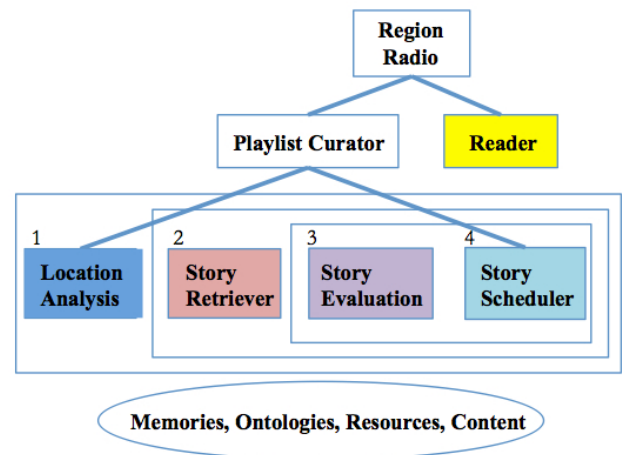


Figure 1: Region Radio high-level architecture

Location Analysis

In the outer loop of the Region Radio planner, the route is analyzed backwards from the goal destination. Place names within a radius of the goal location are identified using a Google Places API (e.g., of type ‘park’ or ‘natural feature’). The proximity radius used in this step is adjusted – if too few places are found in the initial circle, then the radius is increased; if too many place names are found in the initial radius, then the radius is decreased. After parsing through the JSON response to a Nearby Search, we also perform a Place Details Search on each location, to gather its coordinates, town, and state. Under the subsection “Closing the Loops” we describe how the next sub-goal location, again working backwards, is determined.

While we have experimented with semantic web based abilities to find terms that are related to the various place

names found above (e.g., who founded the place; what events occurred at the place), these are not yet integrated into our location analysis. Semantic relationships are found on sites such as Wikidata (e.g., Nielsen, 2013) and can be scraped from Wikipedia, and queries can be augmented with these terms to improve story retrieval.

Story Retrieval

As places are found, and in the future elaborated with semantic webs, stories are retrieved from the Web through a Google API using place-associated keywords. Rather than simply being called once with all available keywords, there is an internal, non-Web search through the space of keyword combinations, and the Google API is called for various subsets of keywords. In the future, semantic web connections will be used to guide the internal search of keyword subsets, but now the process is rudimentary, starting with a most general place name and adding other keywords if the previous search does not return stories of adequate quality. It is worth noting that the addition of even simple meta phrases to a search query (e.g., adding “story about” or “history” to a place name) can increase the density of interesting hits. Note that because there can be multiple place names associated with the goal location in the outer loop, there can be queries for each of these place names during the inner story retrieval loop.

Because we used the Google Custom Search API under a free account, our accesses were very limited in number. In our next phase of development (summer 2019) we plan to upgrade the account as necessary to allow and test more sophisticated and more numbers of Web searches.

Story Evaluation

While the stories are retrieved using place relevant keywords, this is insufficient to ensure place relevance of the results. So post-retrieval, Google Named Entity Recognition finds/tags nouns (e.g., categorized as “LOCATION” and tagged as “PROPER”). The Google geocoder is used to find the distance (e.g., Euclidean) between these locations and the query place. A hierarchical distance measure is also being investigated, but not integrated yet, using information from the Wikidata semantic web that links together places by a variety of relationships, such as inclusivity (e.g., Nantahala Forest surrounds Franklin, NC).

We have also investigated story “interestingness” at some length, but have not yet integrated it into evaluation. Story interestingness is critical so that we don’t end up reading Wikipedia pages, for example, as part of the playlist. Wikipedia and Wikidata are great sources of semantic web material, but often boring as stories. See Future Work for more on our story interestingness investigations. We have (so far) “sidestepped” the need for robust evaluation of story interestingness in Region Radio by biasing our stories to those that appear by existing curations that presumably vet for interestingness (e.g., Conservationist Magazine).

Story Scheduling

Stories from vetted sources that are deemed relevant to places in the vicinity of the (sub)goal location that is the focus of the outermost loop (under Location Analysis), are placed on a priority queue, currently based on relevance scores. The reading time of each story is estimated from the text length and an assumption of an English-speaker reading-aloud rate of 150 words per minute. The estimated reading time is used to estimate the time during the trip when the system must start reading the story aloud (using off the shelf text-to-speech) so that it ends just as the goal location is reached. Our assumption is that listeners would prefer to hear about places that they are approaching rather than places they have passed.

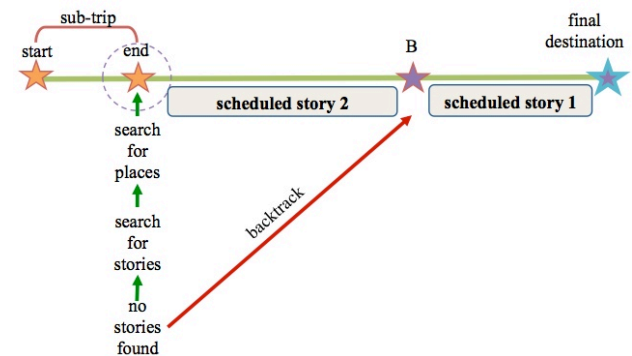


Figure 2: Backwards backtracking search for a playlist

The top rated story for the vicinity of the goal location is placed at the end of the playlist. The estimated starting point of this story becomes a sub-goal location to the planner, which next looks for stories that can be told as this sub-goal location is approached.

Closing the Loops

Figure 2 illustrates an intermediate step in planning of the playlist. The rightmost story was found first (using “final destination” as the goal), the story just to the left of it was found next (using sub-goal B), and its estimated starting point becomes a sub-goal for finding the next, third from the end, primary story in the playlist. Additionally, Figure 2 illustrates that if a third-from-end story cannot be found for the remaining unscheduled sub-trip, then backtracking occurs, and “scheduled story 2” will be retracted and another story drawn from the priority queue for sub-goal B. Backtracking search is a simple, elegant approach, which can be amended by procedures like keeping and tweaking the best schedules found so far in terms of unused time, which is particularly important to avoid repeated backtracking when unscheduled sub-trips become too short.

Other Current Functionality

Region Radio maintains a cache of previously found stories that can be reused as appropriate, currently to reduce

the number of Google Custom Search API calls. This cache will evolve into a repository that is shared across user accounts and be a store for a recommender system.

After retrieval, stories are scraped to eliminate non-story content (e.g., ads), to check for profanity (for eventual warning labels), and replacement of abbreviations (e.g., St. Mary's → Saint Mary's, rather than Street Mary's).

Stories are also introduced with original text that contextualizes the story, orienting the listener to the relevant places (e.g., "fifty miles northeast along US highway 64 is Franklin, North Carolina"). Eventually other contextual and factual information about the place, as can be found on Wikipedia, will be included before launching into a story.

After creation, the playlist stories are read aloud. We have experimented with free text-to-speech readers (e.g., gTTS, IBM Watson, Amazon Polly), with attention to intonation and an ability to adjust reading speed.

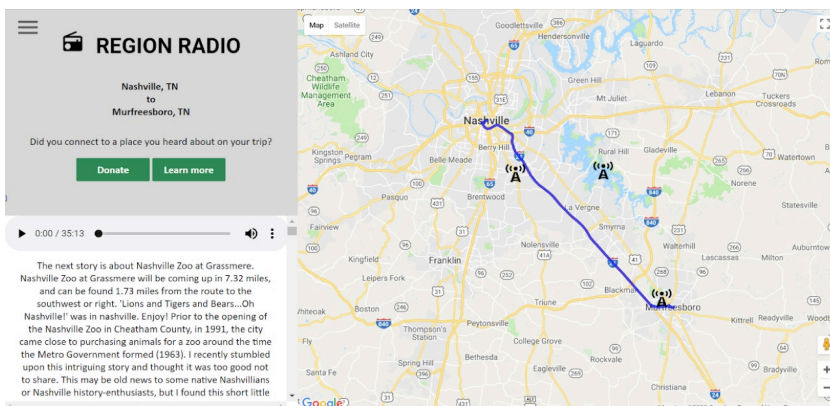


Figure 3: Web-interface to Region Radio

Figure 3 shows the Web interface to the current version of Region Radio, and a sample playlist. A route is shown, along with broadcast tower icons next to the places along the route that are the subject of stories. The first story to be read is about the history of the Nashville Zoo. Two buttons are shown. "Learn more" is functional and links to source files and other related material. The "Donate" button is not yet functional, but is explained shortly.

Ongoing and Future Work

We have mentioned some ongoing and future work (e.g., augmenting queries with semantic web relationships). We are also interested in **evaluating Region Radio** in terms of user satisfaction and whether it increases conservation awareness and place attachment. Long-term effects are hard to evaluate, but we are implementing the potential for donations to the source of stories (e.g., Conservation Magazine) and the subject of stories (e.g., Joyce Kilmer Woods, Jefferson Street Sound). Tracking donations would be a quantitative measure of short-term effects on listeners, and donations might be incentivizing for curators to make their content available to us. The potential for donations

comes with potentially ethical issues, such as guarding against donations to some (e.g., hate groups). In some cases legal agreements with curators are necessary before using them in a public release of Region Radio.

User accounts/profiles are important for recording stories already heard, preference behaviors (e.g. preferences for wildlife versus energy stories), and eventually a support for recommender systems.

The stories in a playlist are currently coupled only by location in the playlist and implicitly by the proximity of the places the stories reference. We are interested in **thematic playlists** (e.g., Civil War sites, Civil Rights sites) using semantic information as mentioned earlier. We have investigated other **measures of story and playlist interestingness** (Ganguly, et al, 2014), which include characterizing narrative trajectories of topics and of sentiments. A playlist that covers distinct topics (e.g., wildlife to human settlement to carbon capture) and sentiments (e.g., habitat destruction to community development) may be more interesting than "flat" trajectories. We've also investigated trajectories within individual stories with sliding windows across the text. Characterizing interestingness is an important functionality that we continue to focus on.

We have investigated **text summarization** (Allahyari, et al, 2017) as a way of previewing and reminding users of story content. In particular, we will integrate story summarizations of one or two lines, which we have satisfactory results for, into the prefacing text to stories in the playlist.

Finally, we will integrate **dynamic capabilities** into Region Radio. Already underway is an allowance for users to swipe past stories, interpreting such actions as preferences, making use of alternate stories in story priority queues, and re-planning the playlist as necessary. Other dynamic capabilities like tracking trip progress and adapting the playlist in response to changes are longer term, after we move to mobile apps.

Acknowledgements

Supported by NSF #1521672 "Collaborative Research: CompSustNet: Expanding the Horizons of Computational Sustainability" and NSF #1623690 "EXP: Bridging Learning in Urban Extended Spaces (BLUES) 2.0." Special thanks to earlier Region Radio development team members, Mateus Winkelmann, John Kim, and Hannah Braun, and to members of the Space, Learning & Mobility Lab. We thank anonymous reviewers for their comments, to include calls for a framework of creative works in which curation might be situated, and for raising the potential for traumatic stories and otherwise out-of-place stories.

References

- Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J.B., & Kochut, K. (2017). "Text Summarization Techniques: A Brief Survey." In International Journal of Advanced Computer Science and Applications(IJACSA), Vol. 8 No. 10, 2017
- Ganguly, D., Leveling, J., Jones, G. J. F. (2014). "Automatic prediction of text aesthetics and interestingness." In Proceedings of the International Conference on Computational Linguistics.
- Greenspan, B. (2011). "The New Place of Reading: Locative Media and the Future of Narrative." In Digital Humanities Quarterly, Vol. 5 No. 3
- Gruenewald, D.A. & Smith, G. A., Eds. (2008). *Place-Based Education in the Global Age: Local Diversity*. Lawrence Erlbaum.
- Hall, R., Shapiro, B. R., Hostetler, A., Collins, H., Owens, D., Daw, C., & Fisher, D. (in press). "Here-and-Then: Learning by Making Places with Digital Spatial Story Lines." To appear in Cognition & Instruction.
- Light, R. (2001) *Making the Most of College: Students Speak Their Minds*. Harvard University Press.
- Madison, M. J. (2011). *Knowledge Curation*. Legal Studies Research Paper Series, Working Paper No. 2011-13. University of Pittsburgh School of Law.
- Maher, M.L., Lee, L., Gero, J. S., Yu, R., Clausner, T., (2016). Characterizing tangible interaction during a creative combination task. Proceedings of Design Computing and Cognition.
- Milligan, M. L. (2011). "Interactional Past and Potential: The Social Construction of Place Attachment." Symbolic Interaction, Vol 21, No 1: pp 1-33.
- Nielsen, F. Å. (2016). "Literature, Geolocation and Wikidata" Tenth International AAAI Conference on Web and Social Media, AAAI.
- Nisi, V. (2017). "The changing panorama of interactive storytelling: a review from locative to transmedia." DOC online - Revista Digital de Cinema Documentário. 43-68.
- Nisi, V.; Oakley, I. & Haahr, M. (2008). Places, location-aware multimedia stories: turning spaces into places. Artech, International Conference on Digital Arts. pp. 72-82.
- Weiss, A. S. (2013). "Exploring News Apps and Location-Based Services on the Smartphone." Journalism & Mass Communications Quarterly, Vol. 90, No. 3. Pp. 435-456.

Opportunities for Computational Creativity in a Therapeutic Context

Lee Cheatley, Wendy Moncur, Alison Pease

University of Dundee, Scotland

Abstract

The question of why and for whom we build creative systems is becoming increasingly relevant. We argue that one potential application area is in therapeutic fields. We investigate the reminiscence practices of 13 bereaved participants; exploring possessions used to support reminiscence; interactions with them, and participants' receptiveness to computational creativity (CC) being used to support them. We use our findings to identify 10 provisional design recommendations for CC in a bereavement context.

Introduction

A decade of increasingly sophisticated CC systems -- and recent developments in other areas of AI (principally research in Constructive Machine Learning) -- has led to impressive generative results in both the arts and sciences, including painting, music, poetry, gaming, drug design, and gene design; usually in collaboration with domain experts. The question of why, and for whom, we develop CC systems, is now ever more pertinent. In (Colton et al., 2015) the notion of a *creativity stakeholder* is raised, as "people who may have something to gain or lose from software which is creative" (*Ibid.* p1). Colton et al. suggest a non-exhaustive list as: "researchers, the wider AI community, funding bodies, experts in the psychology of human creativity, neuroscientists, artists, art critics, journalists, philosophers, educators, the public, and so on." (*Ibid.* p5). In this paper we identify a further (possibly overlapping) community of creativity stakeholder: those for whom the creative process and outcome can have therapeutic value. Creativity can play at least two roles in this context: a created artefact, such as a collage of photos of someone who has died, and the process of the bereaved person putting together the collage, can both be very meaningful in the grieving process. These two roles co-align with the twin strands of research in CC: autonomous creativity in which a system creates an artefact, and co creativity, in which system and person work together.

Creative Arts Therapy (CAT) uses creative experiences to aid people in exploring their feelings. It is used in a variety of contexts, such as helping adoptive parents and child to bond, overcoming conflict, and bereavement; and domains, such as visual arts, dance, drama, music, and poetry. For therapy to be successful, it is necessary to establish a safe space in which the patient feels comfortable and safe in engaging with their feelings. Resources and availability can be an issue, and with CAT there is the

additional challenge of encouraging patients -- who may not think of themselves as creative people -- to be creative. We believe that people for whom the creative process and outcome can have therapeutic value, and associated professions, form an important community of creativity stakeholders, and a novel and intriguing application area for CC systems. In this paper we present our investigations into a subset of this community; the bereaved.

The design of technology to support the bereaved is an emergent theme in Human Computer Interaction (HCI) research (Massimi et al., 2011; Moncur et al., 2015, 2012; Walter et al., 2012). In this study we: (i) identify a new creativity stakeholder group (the bereaved); (ii) employ user-centred methodology to explore current reminiscence practices; ways in which artefacts and possessions are used in reminiscence; and receptiveness to CC bereavement support tools; and (iii) offer a series of provisional design recommendations for CC in a bereavement context. We envisage that such systems could help the bereaved to overcome their grief, continue bonds with those they have lost, and aid therapists in the services they provide.

Related Work

The notion of creativity stakeholders is an emergent theme in CC research which seeks to stress the importance of exploring who CC systems are made for, why, and for what purpose (Colton et al, 2015).

Grief, and the complications which may arise out of it can negatively impact the mental and physical wellbeing of the bereaved, and even increase the risk of mortality (Buckley, 2012; Carey et al., 2014; Mostofsky et al., 2012). Failure to engage with grief can prolong the grief experience, and exacerbate symptoms experienced. Current accepted theories of grief place an emphasis on adaptation to a world without the deceased, and a continuation of bonds with them (Worden, 2009). *Continued bonds* refer to a continued relationship with the deceased. This can be achieved in many ways: writing letters to them; toasts; talking to them; etc. CAT is gaining traction and popularity today as a means of successfully supporting bereaved people. These have patients and therapists explore thoughts and feelings through creative means. For instance, poetry therapy (the writing, and analysing of poetry) has proven useful in processing grief (Shafi, 2010; Stepakoff, 2009; Mazza, 2001).

Recent work in HCI in the context of using technology to support the bereaved has provided design recommendations for and the creation of objects that

memorialise or commemorate the deceased (Banks, 2011; Banks et al., 2012; Gerritsen et al., 2013; Gulotta et al., 2016; Moncur et al., 2012; Odom et al., 2012). Most potential solutions born from these works have been simple memorial artefacts, physical containers holding digital objects. They capture the memory of the deceased statically, and do not foster an evolving relationship with the deceased but a continuation of what once was. The only examples of less static memorialisation and systems that continue bonds have come in the form of systems using artificial intelligence to mimic those now dead – which have been met with mixed reception. The most well-known of these is the askroman chatbot, created by Eugenia Kuyda, intended to reply in the same way as the person she had lost (Newton, 2016).

Method

Our goal was to identify a series of design opportunities for CC support systems through exploration of participants' possessions related to two people they had a relationship with- one alive (subject A) and one deceased (subject D): participants interactions with possessions, and how these changed dependent on the materiality (whether possessions were physical or digital), and whether the subject of reminiscence was subject A or D. We also explore receptiveness to and preferences for CC options to help the bereaved continue bonds with the deceased.

Approach

Semi-structured interviews were conducted on a one to one basis. Interviews took place at the participant's home to ensure they had access to their possessions. When a home interview was not possible, interviews were conducted in a private meeting room at the university, or via audio and video conferencing software. The interviews were split into three sections. **Section One** explored participant's possessions related to an alive subject of reminiscence (subject A), participants interactions with possessions: the possession participants valued most, why, cues that led to interaction, and the impact of the interaction; the materiality of possessions was also discussed, and participants explored whether they would feel the same about their possessions if they were physical rather than digital and vice versa. **Section Two** was the same as the first section but in relation to a deceased subject (subject D) with the addition of questions exploring how participants interacted with possessions differently since the subject had died, and how these interactions and feelings differed from those in section one. **Section Three** asked participants how they felt about peoples' possessions outliving them and invited them to explore what they would be comfortable with being used as input for a potential computationally creative system (if anything). Two examples were used to illustrate how CC could be used: a poetry generation system with user input, and an image generation system with user input. These examples were chosen as two of the most accessible forms of creativity.

Participants

For inclusion in the study, participants had to be aged 18-33, speak English, be computer literate, live in the UK, and have been bereaved for between 6 months and 7 years. This was to ensure participants were not at their most vulnerable and increase the likelihood of those they had lost having had a digital presence. No exclusion criteria were set for gender, possessions, subject of reminiscence, or cause of death. Participants almost exclusively spoke about family members or partners who had died of natural causes. Most of the deceased referred to were elderly, and only two participants spoke of someone who frequently interacted with technology. One participant spoke about a friend that died - this was also the only subject that died by suicide. The perspective of the data gathered, and the subsequent design opportunities identified, have been influenced by this demographic information.

Thirteen participants were recruited (8 female, 5 male) through posters in university campuses, counselling services, public libraries and museums, and churches, as well as through social media sites, and a webpage set up for the study. They were anonymised via assignment of acronyms (P1 – P13). The time since bereavement occurred ranged from 1 to 7 years, and all participants indicated they were close or extremely close to the people that they spoke about.

Ethics

The institution of the authors granted ethical approval for the research. Procedures were formulated to minimize risk for participants and the interviewer, due to the personal and sensitive nature of the interviews. The interviewer completed a mental health first aid course offered by the National Health Service (NHS) to better ensure they were able to provide support and guidance. Details for free counselling and support services were available if needed. The interviews were audio recorded and transcribed.

Analysis

Thematic Analysis (Braun, V. & Clarke, V. 2008) was employed to analyse the interview transcripts. Data was grouped into themes (coded) and analysed iteratively to refine themes across all participants. NVivo, qualitative analysis software, was used.

Results

Five key themes surfaced from the data: (1) possessions, and their properties (2) interactions with possessions, (3) privacy and permissions, (4) contrasts in interactions with, and properties of possessions, and (5) receptiveness to technological solutions. We discuss each in turn here.

Possessions, and their properties

Our participants spoke of physical possessions such as photographs, letters, clothing, books, and jewellery, and of digital possessions such as photographs, emails and other text-based archives, in game gifts, playlists, and eBooks.

Possessions related to subject A were mainly digital and possessions related to subject D for all, but two participants were mainly physical –as they spoke of someone they had lost that had not actively engaged with social media etc. Every participant favoured physical possessions related to subject A mainly due to their tangibility, sense of history, and the fact some were created by the deceased which left a personal mark such as handwriting. The lack of space digital possessions require, and the level of access they offer to participants was reported as positives by participants. P12 summarised this in the following: *“I’m just glad all the emails aren’t physical because I’d never have all the space for them.”*

For possession related to subject D the same as above was true for all but two participants. One of whom only had digital possessions, and as such valued those without the ability for comparison. The other had both physical and digital possessions and preferred the digital as they were harder to lose (easier to backup), and as they interacted with the person they lost through social media looking back on those messages made them feel closer to the person.

Interactions with possessions

Participants interacted with possession related to subject A for a number of reasons: when they were feeling down and wanted cheered up; out of necessity (e.g. photographs on walls and as screensavers); when they wanted to remember something; or when something made them think of the person or the event the possession related to. These interactions were frequent and not seen as special unlike interactions with possessions related to subject D. In the immediate aftermath of loss participants interacted with possessions related to the deceased much more. This interaction would decrease as participants came to terms with their loss. By this point interactions become infrequent but more meaningful than those with possessions related to the alive subject. Interactions with possessions related to subject D were brought about mainly by anniversaries, special occasions, or necessity (e.g. moving house). Interactions, much like those for possessions related to the subject A, brought to mind happy memories for participants but also a sense of longing. Interactions were bittersweet.

Privacy and Permissions

Participants (n = 6) were concerned about their privacy and that of the deceased. Some participants felt digital possessions afforded them a level of privacy in public that physical possessions don’t. P1 stated: *“The phone is there with me...It’s quite nice and private, people don’t really know what you’re looking at. It keeps it personal, between you and the person.”* Despite this they were worried about the privacy of possessions available online as can be seen in P1’s words: *“...I’m scared it’s going to be out there for everyone.”* This was most strongly felt in relation to the use of possessions related to subject D in the creation of new possessions to help memorialise or continue bonds. Both in regard to the privacy of the deceased, and the

bereaved. Participants were worried about who could use what possessions, who could access the possessions, and whether some things should be shared. P13 felt it was all about the context, if a possession is shared then it can be used, and that if you have access to a possession it can be used.

Contrasts in interactions with and properties of possessions

Digital and physical possessions

Participants preferred physical possessions but interacted with digital possessions more, especially in relation to someone that is still alive, or someone who had a digital presence but is now deceased. This was down to two key things: 1) easier access to digital possessions than physical possessions; and 2) the privacy afforded to digital possessions viewed on a private screen. People know you are looking at a phone, but not what is on the phone. Despite this, interactions with physical possessions are seen as more impactful, in part due to infrequent interactions as noted by P4: *“...it’s more valuable because I’m not interacting with it every moment of the day.”* Participants liked the sensation of actually connecting with a possession. Being able to feel or smell a possession. They liked that these possessions could degrade over the years or through frequent interaction. P2 felt physical possessions were more *“precious”* due to their potential to degrade.

Between possessions related to subject A, and D

Participants interacted with possessions related to subject A more than subject D. They do so to cheer themselves up or to reminisce and cues are more frequent as these possessions are embedded in their daily life. Whereas possessions related to subject D take on an increased sense of value and sentimentality and are stored away safely - to protect the possessions, and also to insulate the participant from the possession. Interactions with these possessions brings a sense of longing, loss, or finality alongside the happy memories. Participants noted they interacted with possessions related to the deceased with an increased frequency after their passing, which would gradually decline as they came to terms with their loss.

The properties associated with physical possessions that made participants favour them to digital (personalisation, hard work, etc.) in the case of the deceased subject gave interactions with these possessions a feeling of intimacy, and a sense of continued bonds with the person – almost a feeling as if they were still there with them as was the case with P8 for example: *“...the cardigan is the most important because it’s like a sense she’s with me...”* This feeling of continued bonds was also evoked by digital possessions. P5 spoke of someone who had a large digital presence, stated they preferred digital possessions because: *“...I think there’s a lot more depth to the music and messages I have online because they’re a lot more recent as well...”* Before going on to add they made them feel closer to the person they had lost.

Receptiveness to technological solutions

Despite some participants having reservations, all participants were open to the use of CC in a bereavement context with P13 going as far to say they had experienced problems with memorialisation they felt could be tackled by a CC system. 10 participants liked the idea of a system that could create new, reflective, possessions from newly created input or old social media posts or photographs to support memorialisation and reflection. P8 and P13 felt systems should provide the option to collaboratively use such a CC system. They indicated the use of such a system would be highly contextual and only as the bereaved and or deceased found acceptable. Additionally, participants felt it important the privacy of not only the bereaved, but the deceased was protected. Participants did not want systems that altered or destroyed the input or that mimicked or imitated the deceased. They (n = 6) feared the loss of their possessions, and that their memories or the reputation of the deceased could be tarnished. Whilst the majority (n = 10) of participants spoke positively about these potential systems, two viewed them as clinical. Disconnected from the people and relationships and were worried they would not be able to accurately depict how they felt, or the relationship they had.

Provisional Design Recommendations

In this section we present the 10 provisional design recommendations for CC bereavement support tools that arose from the study and briefly discuss them. A CC bereavement support tool should:

Be available freely online – Support is not always available to those who need it and when it is it is not always affordable. The provision of a free supplementary support tool would ensure as many people as possible who need help could access some form of help.

Output physical and digital possessions – Not only to support user preference in terms of tangibility but also support required levels of interaction. Participants interacted with digital possessions more but favoured physical, whilst they interacted with possessions related to the deceased more in the immediate aftermath of loss and gradually as they came to terms with their loss interacted with these possessions less. Which could suggest the provision of digital possessions in the immediate aftermath of loss and physical possessions later could be beneficial.

Present framing information – To increase user understanding of the possession and their impact on its creation which could contribute to increased feelings of ownership over possessions created with the system and thus the value attributed to the possession.

Incorporate degradation into digital output – Degradation contributed hugely to the value attributed to physical possessions and likewise to the meaningfulness of interactions with them. This could be replicable in digital possessions and lead to the creation of more valued digital possessions with which interactions are more meaningful than with ordinary digital possessions.

Require users participate in creation process – Participation in the creation process may support users interact with their grief and lead to the creation of meaningful possessions.

Allow for a varied source of input – to allow users to express their feelings in whichever way they feel comfortable or proficient in. Additionally, the option to utilise pre-existing input such as social media posts would allow people to avoid interaction with possessions related to the deceased at times they do not wish to interact with them and when it may negatively impact them to do so.

Employ sentiment analysis – Carried out on user input sentiment analysis could create possessions reflective of how users feel which could help users reflect on how they're feeling and to continue bonds with those they have lost. Additionally, the personalisation afforded by reflective output could increase the value attributed to it and the creation of a personal connection.

Allow for and foster repeated use – The provision of reflective output to frequent users could help chart the user's bereavement journey and show they are coming to terms with their loss or indicate when they may need to seek additional help.

Allow private and collaborative creation – To ensure those who wish to grieve alone can do so and reflect on their loss individually, and that those who wish to grieve in company with likewise bereaved people can do so together and share stories.

Be secure and private – Input and output should be available only to the person or people who wrote it and those they wish to share it with. To protect their confidentiality and privacy, and to ensure they trust the system.

Conclusions and Future Work

Participants were open to the use of CC to help memorialise or continue bonds with the deceased albeit to different degrees. Many were enthusiastic, and some felt these systems could have helped overcome problems they have faced already. Despite this, there were reservations. Participants had some misconceptions about artificial intelligence and CC. They worried it meant systems designed to support the bereaved will seek to replace and mimic the deceased, rather than provide an interactive process that helps the bereaved interact with their grief and reflect on their relationship with the deceased and their own bereavement journey. We identified 10 provisional design recommendations for the design of CC systems to support the bereaved and provided an overview of these opportunities. We will explore and test these recommendations in future studies.

Acknowledgements

We thank our participants for sharing their time and experiences with us. We acknowledge the financial support of an EPSRC Doctoral Training Award, and the European Commission for the H2020 Council of Coaches project, under Grant Agreement No. 769553.

References

- Banks, R., 2011. The future of looking back, Microsoft Research series. Microsoft Press, Redmond, Wash.
- Banks, R., Kirk, D., Sellen, A., 2012. A design perspective on three technology heirlooms. *Human-Computer Interact.* 27, 63–91.
- Braun, V. & Clarke, V. 2008. Using thematic analysis in psychology, *Qualitative Research in Psychology*, 3:2, 77-101
- Buckley, T., Sunari, D., Marshall, A., Bartrop, R., McKinley, S., Tofler, G., 2012. Physiological correlates of bereavement and the impact of bereavement interventions. *Dialogues Clin. Neurosci.* 14, 129.
- Carey, I.M., Shah, S.M., DeWilde, S., Harris, T., Victor, C.R., Cook, D.G., 2014. Increased Risk of Acute Cardiovascular Events After Partner Bereavement: A Matched Cohort Study. *JAMA Intern. Med.* 174, 598.
- Colton, S., 2011. The painting fool in new dimensions, in: *Proceedings of the 2nd International Conference on Computational Creativity*.
- Colton, S., Pease, A., Corneli, J., Cook, M., Hepworth, R., & Ventura, D. 2015. Stakeholder groups in computational creativity research and practice. in TR Besold, M Schorlemmer & A Smaill (eds), *Computational creativity research: towards creative machines*. vol. 7, Atlantis Thinking Machines, vol. 7, Atlantis Press, Amsterdam, pp. 3-36.
- Gerritsen, D.B., Tasse, D., Olsen, J.K., Vlahovic, T.A., Gulotta, R., Odom, W., Wiese, J., Zimmerman, J., 2016. Mailing Archived Emails as Postcards: Probing the Value of Virtual Collections. *ACM Press*, pp. 1187–1199.
- Gulotta, R., Odom, W., Forlizzi, J., Faste, H., 2013. Digital artifacts as legacy: exploring the lifespan and value of digital data, in: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 1813–1822.
- Hanson Robotics, 2017. Website. Available at: <https://goo.gl/McnrWt>
- Massimi, M., Odom, W., Banks, R., Kirk, D., 2011. Matters of life and death: locating the end of life in lifespan-oriented HCI research, in: *Proceedings of the Sigchi Conference on Human Factors in Computing Systems*. ACM, pp. 987–996.
- Mazza, N. *Journal of Poetry Therapy* (2001) 15: 29.
- Moncur, W., Bikker, J., Kasket, E., Troyer, J., 2012. From death to final disposition: roles of technology in the post-mortem interval, in: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 531–540.
- Moncur, W., Julius, M., van den Hoven, E., Kirk, D., 2015. Story shell: the participatory design of a bespoke digital memorial, in: *Proceedings of 4th Participatory Innovation Conference*. pp. 470–477.
- Mostofsky, E., Maclure, M., Sherwood, J.B., Tofler, G.H., Muller, J.E., Mittleman, M.A., 2012. Risk of Acute Myocardial Infarction After the Death of a Significant Person in One's Life. *Circulation* 125, 491–496.
- Newton, C. 2016. Speak, Memory. *The Verge*. Available at: <https://goo.gl/g6zmi6>
- Odom, W., Banks, R., Kirk, D., Harper, R., Lindley, S., Sellen, A., 2012. Technology heirlooms?: considerations for passing down and inheriting digital materials, in: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 337–346.
- Shafi, N., 2010. Poetry therapy and schizophrenia: Clinical and neurological perspectives. *J. Poet. Ther.* 23, 87–99.
- Stepakoff, S., 2009. From destruction to creation, from silence to speech: Poetry therapy principles and practices for working with suicide grief. *Arts Psychother.* 36, 105–113.
- Walter, T., Hourizi, R., Moncur, W., Pitsillides, S., 2012. Does the internet change how we die and mourn? Overview and analysis. *OMEGA-J. Death Dying* 64, 275–302.
- Worden, J.W., 2009. *Grief counseling and grief therapy: a handbook for the mental health practitioner*, 4th ed. ed. Springer Pub.

WanderGAN

Eyal Gruss

eyalgruss@gmail.com

Abstract

A search journey in the imagined visions of a neural network. Given a photo, an artificial intelligence painter tries to recreate its likeness, and takes us through a visual journey in its search for the perfect reproduction. The spectator can intervene in the process, and focus on areas of her interest in the intermediate imagery. The painter will then shift its efforts to recreate the chosen impression. The emerging experience may resemble wandering within a vision or a dream.

Introduction

Modern generative neural networks can produce mesmerizing visual outputs ranging from psychedelic to photorealistic to artistically creative imagery. But how can these be used to tell a story and create an immersive experience? BigGAN (Brock, Donahue and Simonyan 2018), is a recent state-of-the-art generative adversarial network capable of generating diverse images of high-fidelity (Figure 1). Artist Mario Klingemann¹ has manually searched or found interesting non-realistic images generated by the algorithm, and created ad-hoc fiction (Figure 2).

BigGAN is a decoder – it generates images based on a latent list of numbers. It lacks a corresponding encoder – i.e. there is no structured way to find the numbers that would generate a desired image. Trying to find these numbers is known as the inverse rendering problem. In this work, we use evolutionary search methods² to solve this problem. By iterative trial-and-error we find the numbers that generate images more and more similar to our target image (Figure 3). The final results can vary from a perfect match (Figure 3A), a result of similar content and style (Figure 3B), and a result of similar semantics (Figure 3C). The search process constitutes our journey.

Two different metaphors could be used to describe the complexity of the problem. In the first metaphor, we are traversing a foreign world, in order to find a specific artifact or viewpoint of unknown location, having just a photo of it. But, instead of being able to go north, south, east or west, we have 2000 different possible directions to go. In the second metaphor, we have a savant painter who

is blind and deaf. We try to get him to paint a specific image, by only giving him cues of "hot or cold". In both cases a visual journey is created by the process of searching for the desired image.

In WanderGAN, we visualize this search process and allow experiencing it as a visual journey³ (figure 4), as well as interactively intervening in the choice of goal. Sometimes in a dream we try to follow an elusive visual stimulus. Sometimes when daydreaming or recollecting a dream, we try to recall a mystical place of which we have a vague impression. WanderGAN tries to simulate such experiences, and the spectator may realize that the journey is actually more important than the goal. However, it is the search for the goal that creates the journey. This works condones the search that leads the marvelous journey to one's Ithaka.

WanderGAN can be experienced in two different modes: (1) An interactive installation – the work is exhibited on a large touchscreen that the spectator interacts with for wandering and creating her own story. The installation can be adapted to be site-specific, by using images from the location, as the images opening the journey. (2) A live video-art performance by the creator.

On the technical side, we face three main challenges: (1) finding an efficient global optimization method to solve the high-dimensional (1128 dimensions) non-linear search problem of inverse-rendering, which we addressed by combining evolutionary strategies, local search, and pre-trained image classification. (2) defining a relevant similarity metric to balance between visual similarity and perceptual similarity, that works well also when the images being compared are very dissimilar, such that misaligned images of similar semantic and stylistic contents are preferred over other dissimilar images (3) an efficient implementation allowing usage in the capacity of a live interactive art installation.

References

Brock A.; Donahue, J.; and Simonyan, K. 2018. Large Scale GAN Training for High Fidelity Natural Image Synthesis. arXiv:1809.11096 [cs.LG]

¹ <https://twitter.com/quasimondo/status/1064248673683554305>

² See video: <https://youtu.be/DgKhdOJmTpE>

³ See video: https://youtu.be/-oFv_j3gtIE



Figure 1: Photorealistic images generated with BigGAN from (Brock, Donahue and Simonyan 2018).



Figure 2: Non-realistic Images generated with BigGAN by Mario Klingemann.







Target image	Most similar generated image
	 <p style="text-align: center;">A</p>
	 <p style="text-align: center;">B</p>
	 <p style="text-align: center;">C</p>

Figure 3: Examples of WanderGAN's evolutionary search results: A=perfect match B=content and style match C=semantic match.



Figure 4: Illustrative image from WanderGAN installation for Midarom Festival, Israel 2019.

Pictures of J** Girls in Synthesis

Eyal Gruss
eyalgruss@gmail.com

Ayelet Sapirshtein

Vered Heruti
Hamidrasha, Beit-Berl College
Kfar Saba, Israel

Abstract

Pictures of Jap Girls in Synthesis^{1,2,3}, is a machine that creates live, visual poetry (Figure 1). The machine reads, or listens to poetry in a variety of languages. It synthesizes in real-time, visual images illustrating its understanding and interpretation of the text. As an interactive installation, poets can perform live poetry, with the machine simultaneously translating it into visual imagery, allowing a synesthetic and a more universal experience (Figure 2).

Introduction

As poets and digital artists our main forms of expression are based on text, either human words or computer code. We are interested in how to augment our words, give them a place in space, and realize them in other mediums. How can we visualize poetry? How can we define and create visual poems? How can we allow speakers of different languages to share a universal poetic experience? (Figure 3) How can we create a synesthetic experience for poetry, involving multiple senses, and combining words, sounds, space, and visual imagery into a unified experience? Can we automate this process? We call it synthetic synesthesia.

To quote Mitchell: "Word and Image" seems to be better understood as a dialectic trope... It is a dialectical trope because it resists stabilization as binary opposition, shifting and transforming itself from one conceptual level to another, and shuttles between relations of contrariety and identity, difference and sameness. Mitchell's statement about the relations between image and text can also be connected to this project. Mitchell (1996) formulates these relations through one welded word – "imagetext". Mitchell objects the perception of image and text as binary opposition con-

cepts and claims that there are similarities as well as differences between the two terms.

On the technical side, we are using state-of-the-art deep learning algorithms for speech-to-text⁴, language translation⁵, and the AttnGAN⁶ for text-to-image synthesis (Xu et al. 2017). While the results are far from photorealistic, we get images of surrealistic quality, containing colors, shapes, textures, patterns and objects related to the words and their context, and creating a poetic experience. This project was created in a 30-hour hackathon, at Geekcon-Art⁷ Hamidrasha 2018.

References

- Mitchell, W.J.T. 1996. Word & Image. In Nelson, R.S. and Shiff, R., eds., *Critical Terms For Art History*. Chicago: The University of Chicago Press, 51–61.
- Xu, Z.; Huang, Z.; Gan, H.; and, He, X. 2017. AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks, arXiv:1711.10485 [cs.CV]



Figure 1: Generated image for the phrase from David Bowie's Ashes to Ashes, for which the work is named after.

¹ The title of this extended abstract has been changed at the request of the ICCO Organising Committee.

² "Pictures of Jap girls in synthesis" is a quote from David Bowie's Ashes to Ashes (1980).

³ See project videos: <https://youtu.be/bGAx9wyF8Ks> (short), <https://youtu.be/4zdo-bPXszY> (full).

⁴ <https://cloud.google.com/speech-to-text>

⁵ <https://cloud.google.com/translate>

⁶ <https://github.com/taoxugit/AttnGAN>

⁷ <https://geekcon.org/art2018>



Figure 2: Savyon, an Israeli poet, performing one of her poems in Hebrew, live with the interactive installation.



Figure 3: An example of verse in Farsi. While we do not understand the text, the visual image may be universal.

Algorithm as Determinant of Form in Music

Halley Young

Computer Science
Department University of Pennsylvania
Philadelphia, PA USA
halleyy@seas.upenn.edu

Abstract

In this creative submission, algorithmic processes are explored as ways of establishing form in music.

Introduction

In addition to having different harmonies, instruments, and rhythmic styles, different cultures are associated with different ways of organizing music. In the 17th/18th-century common practice style, there are well-known conventions for developing both small-scale and large-scale patterns of sound; for instance, the antecedent-consequent structure of the period establishes order on the range of 5-10 seconds, while sonata form is a highly involved formal device for ordering entire movements (or even suites of movements) (Caplin 2000). The 20th and 21st centuries are associated with new formal devices, including process-based music in which events occur such as “reaching a limit beyond which the preceding process cannot continue” or “the arbitrary stopping of a process” (Tenney 1973). The processes used in this way can often be described mathematically, such as the use of rhythmic cannons by Messiaen (Carl Bracken 2009).

Simultaneously, since at least the 1950’s, another type of sonic art has become popular - that of sonification. Sonification is the practice of converting data to sound (Fry 2005). It has been described by some as having actual scientific value for apprehending patterns in scientific data. However, its status as music is up for vigorous debate, as the title “Sonification \neq Music” in the Oxford Handbook of Algorithmic Music suggests (Scaletti 2018). This has to do with the fact that the most literal sonifications often willfully ignore the principles of what a Western audience finds pleasant (as well as features which are arguably more universal) in favor of keeping the meaning as explicit as possible.

I am interested in work at the intersection of these two areas. Specifically, I am exploring how to structure music using algorithmic processes. The distinction between this enterprise and traditional explorations of form is that when form is determined by a computational process, it might take a shape that was not envisioned (or even imaginable) to the composer. However, this enterprise is also different

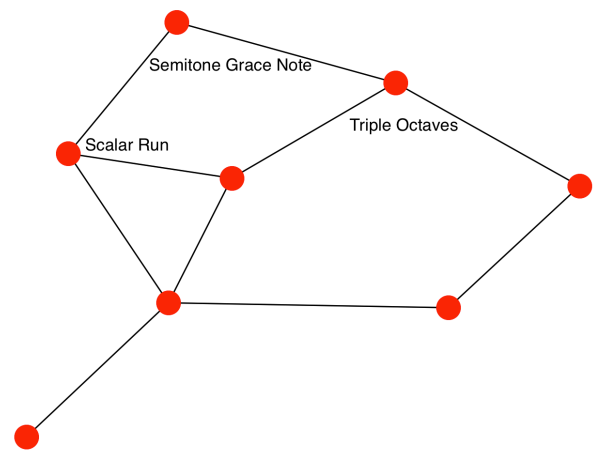


Figure 1: The structure underlying the convolutional graph network composition

from traditional sonification, in that I explicitly want to generate output that is heard as music, which translates into a much less literal interpretation of the algorithmic process.

Sonification of Convolutional Graph Networks

One of the most recent trends in deep learning is Convolutional Graph Networks (Defferrard, Bresson, and Vandergheynst 2016), which allows for the application of neural networks to data structures in the form of graphs. While I highly suspect that in the future music will be made through the usage of CGN’s, here I attempt to illustrate the essential functioning of CGN’s - how they “smear” the vector values of each node around the graph progressively through local averaging of the nodes. The graph structure illustrated in Figure 1 is sonified. Each node is assigned a different instrument and a different musical idea. The musical ideas are each not traditional motifs, but rather ideas that can be implemented to varying degrees - for instance, it is possible to play none, half, or all notes with a semitone-grace-note before it. At each time step, the value of each node is updated by the mean of the value of its neighbors - for instance, the node that is associated with semi-

tone grace notes will to also be affected by the ideas of scalar runs and triple octaves, which are associated with two neighboring nodes. As the activation spreads, musical ideas become more homogeneous.

Sonification of Convolutional Graph Networks

In the second example of such a piece, I try to describe in music the execution of a short imperative program. This is “program music” in two senses - it fits the traditional definition of “program music” in that it is defined by a narrative, but the narrative is a piece of code rather than a natural English story! I invented ways of representing the specific concepts seen in an imperative program. For instance, I model a “while-loop” by repeatedly changing a motif, but stopping the process only when the final note finally changes. Motif transformations represent function application on variables. Under the melody is a constant, fast-paced “clock”, which is ticking both before the program initiates and after it terminates.

Figure 2 shows the code of the sonified algorithm.

```
x = f()
y = g()
h = stutter(x)
g = stochInvert(y)
while (g[-1] != "D") {
  g = perturb(g)
}
z = h()
for (i = 0; i < 2; i++) {
  bool_var = "G"
  if (z)
    bool_var += 1
  print(bool_var)
  z = h(z)
}
x = perturb(x)
print(x[-1] == "F#")
exit()
```

Figure 2: The code of the sonified algorithm

Sonification of Towers of Hanoi

The Towers of Hanoi is a classical game involving rings being moved between pegs which is used to teach about recursion. The way to optimally play is simple to describe if you know the algorithm, but very difficult to arrive out without that knowledge or the ability to carry out the recursive procedure it involves. The algorithm has some symmetrical and some asymmetrical qualities - while at every level the task is to move the top $n - 1$ rings twice, moving the n th ring once in the middle, the exact order of what is being moved there follows an interesting fractal-like pattern. As such, it became an excellent target for algorithmic composition. I assigned the flute a different melody at every time step according to the type of movement, and con-

tracted or expanded each time step according to the size of the ring being moved. The instrumentation of the rest of the instruments was related to the teleological movement being effected at higher levels due to the actual movement (e.g., the ring of size 4 was being moved to the right in order to move the ring of size 3 to the left, which was done in order to move the ring of size 2 to the right, etc.) In post-processing, after running the Python script, I changed the dynamics and some of the pitch material of the inner voices (while keeping timbre, attack points and rhythm intact), which gave it more of a directed feeling than would be possible otherwise using such an algorithm.

References

- Caplin, W. 2000. *Classical Form: A Theory of Formal Functions for the Instrumental Music of Haydn, Mozart, and Beethoven*. Oxford Press.
- Carl Bracken, Gary Fitzpatrick, N. M. 2009. Tiling the musical canon with augmentations of the ashanti rhythmic pattern. *Bridges: Mathematics, Music, Art, Architecture, Culture*.
- Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In Lee, D. D.; Sugiyama, M.; Luxburg, U. V.; Guyon, I.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc. 3844–3852.
- Fry, S. P. 2005. A brief history of auditory data representation to the 1980s. In *First Symposium on Auditory Graphs*.
- Scaletti, C. 2018. Sonification \neq 6 music. *Oxford Handbook of Algorithmic Music*.
- Tenney, J. 1973. Form in 20th century music. *Dictionary of Contemporary Music*.

Hidden Worlds

J. Rosenbaum

Department of Art
RMIT University
Melbourne, Victoria, 3000 Australia
jr@jrosenbaum.com.au

Abstract

Hidden Worlds is a series of artworks created by Machine Learning exploring gender through the lens of computer vision. The works are first created by a DCGAN, then responded to by J. Rosenbaum before being reinterpreted into a generative narrative using image classifiers and tied together using Augmented Reality. This collaborative series of works is part of Rosenbaum's ongoing work with machines using 3D Modeling, machine learning and Augmented Reality.

Introduction

Hidden Worlds uses Artificial Intelligence and Augmented Reality to examine gender through the lens of computer vision. These works use a computer trained on Greek and Roman statuary to generate its own which I interpret in my own way. I use another AI to write descriptive content for each work and generated music to create a multi-media interactive installation.

Can computers see gender? Without being trained in traditional binary notions of gender what can they produce? And how do we interpret the results? J. Rosenbaum presents Hidden Worlds, a series of Artificial Intelligence Computer Generated artworks using mobile Augmented Reality technologies to see gender through the lens of computer vision. A Neural Network that has been trained in thousands of images of Greek and Roman statuary attempts to create its own. Rosenbaum then takes the output and seeks to find the truth inside the computer generated work and reveal that to the viewer. Another Neural Network looks at the works and attempts to write poetry based on what it sees. This has been incorporated into a soundscape inside the app. Viewers will see the computer generated images and watch them come to life inside the app as the work is transformed and reinterpreted by human eyes and hands. The language will be alien, computer driven showing a collaborative effort between human and machine. This experimental work invites questions about computers creating art, about how machines see humans and gender and idealized beauty.

Project Development

This series works with a small dataset, ~4000 images of Greek and Roman statuary. The nature of curation and critique is a constant recurring theme, the artist curates a dataset of thousands of images for the machine to work with. The discriminator critiques the results of the generator, I curate and critique the results that passed the discriminator.

Once the dataset was complete, I started the weeks of training of the neural network. I would load the samples as they emerged from in the latent space of training and watch as it learned. Sifting through these images for ones that inspired me was another exercise in curation. and I watched as the concept evolved in my head. Being a small dataset, the results were still abstract, but open to interpretation. The generated works were abstract as expected but with recognizably human shapes. As the dataset was extremely idealized it was a delight to see exaggerated bodies and delightful curves emerge, works beyond what I had expected to find. I delighted in the textures and the shapes that I saw emerging.

I selected my favorites and made drawings of what I saw in the works, the figures emerging within the abstract forms. from those drawings I made 3D models and from those models I made augmented reality interactions. While close you see something abstracted but as you move back it resolves into something more tangible and real. As with pointillism and cubism you can appreciate the details close, it's only until you step back and take in the entire work that the forms start to resolve into something more than the sum of their parts. It also reflects the training process of the works, the slow coming in to being of certain forms. Some final artworks stood well with just the drawings saved as animations, and some felt best with 3D models attached.

The images and the different 3D models were submitted to the narrative writer which created passages of generated text. I did some minor grammatical edits on the texts, and had a text to speech synthesizer speak the words. It felt important to have digital voices as the narrators of these works and I feel particularly pleased with how the gender of

the subject flips around seemingly at random. the gender of the people in these works feels fluid and I am obscurely proud the machine saw and automatically produced these gender fluid results.

From here I brought them all together in unity, the spoken word, music based on the generated words and the interactions, captions and options to turn off the sound were added and submitted to the different app stores.

This series is a true collaboration for me between my computer and my self. The art is generated by machine with the information I provided it, I then worked back into its results and submitted them back to the machine to see its interpretation of my interpretation. Back and forth we would go, creating work, creating art based on that work, a mini collective of two.

Technological Development

I constructed Hidden Worlds by chaining together different machine Learning projects and training them in my own specific datasets to get the results I was after. this is an initial experimental round in the development of a greater project around viewing gender through the lens of computer vision. The concept behind this was to take the idealized figures of Greek and Roman statuary and see how a DCGAN blends them together. I have selected these works from the latent samples during the creation process, I wanted something more abstract, open to interpretation. from there, I created 3D models as an interpretation of the GAN's output.

The DCGAN (Radford et al. 2015) I used was by Taehoon Kim on the TensorFlow framework with some customizations by me and my own dataset. This is an easy GAN to get started with and I enjoyed working with it.

From there I worked with Ryan Kiros' Skip Thoughts (Kiros et al. 2015) and Neural Storyteller to create captions for the works. using MSCOCO Image recognition and captioning (Lin, Tsung-Yi et al. 2014), neural storyteller creates a small story. For me, the main success was in the abstract thoughts it generated and the way that gender was a completely fluid concept. this writing is inherently non-binary in its construction and is ideally suited to my work.

The voices were made using Apple's own voice synthesis and the music tracks were made in LangoRhythm¹—a midi piano music writer that assigns note values to letters and durations to vowels and the way they occur to create interesting musical compositions. Each musical track is based on the generated captions.

These works have been woven together in Unity to create an Augmented Reality Experience².

Conceptual Statement

Conceptually this work explores the nature of gender and how computers construct and see gender. Working with a DCGAN to create the works, then working back into them myself before submitting them to a classifier and writer shows me machine perceptions of gender when it comes to

trans and gender non-conforming bodies. This series is also an exercise in working collaboratively with my machine, gaining inspiration from the machine output to create a new work then working with the machine definitions.

The app³ is the glue behind this work, it holds all of these disparate concepts together and entwines them, showing the collaborative process between human and machine in interactive sculptures and sketches.

While these works literally transform when viewed they are about the internal dialogue of transness and of hearing yourself correctly or incorrectly gendered. That disconnect that is felt when a pronoun that doesn't fit is used and the comfort of the correct pronoun. These works seek to create this discomfort in people who have never had cause to question the gender they were assigned at birth and hopefully challenge assumptions around assigned pronouns and pronoun assumptions.

These works use multiple systems, none of which were designed for the express purpose I put them towards. They are tied together in a cohesive exhibition and application interlacing the narrative with the human created artworks and the machine generated artworks. Using a neural network designed for faces to generate bodily works, using a classifier that isn't trained on gender and a romance dataset shows that even without specifically fine tuning the datasets and the classification training I can produce something that makes us question gender. This all ties into my research exploring computer perceptions of gender. In this case it is clear that these works do not elicit a binary reaction from the classifier and the pronouns from the narrative writer switch, seemingly at random.

My work is intrinsically about gender. I explore how computers view and create gender and how we respond to gender that is outside the binary. I am non-binary and my work reflects living in a binary world as a non-binary person. Machine Learning is a fascinating way to explore this subject as it is inherently binary in nature, but knowing only what it has been taught or shown it can be as biased as humanity or completely unbiased, creating amazing gender mashups and diverse gender explorations.

References

- Kiros, R.; Zhu, Y.; Salakhutdinov, R.; Zemel, R.S.; Torralba, A.; Urtasun, R.; and Fidler, S. 2015. *Skip-Thought Vectors*. arXiv:1506.06726 [cs.CL]
- Lin, T.-Y.; Maire, M.; Belongie S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C.L.; and Dollár, P. 2014. Microsoft COCO: Common Objects in Context. arXiv:1405.0312 [cs.CV]
- Radford, A.; Luke M.; and Soumith C. 2015. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. arXiv:1511.06434 [cs.LG]

³ To download the app and see images from the project <https://www.jrosenbaum.com.au/projects/hidden-worlds/>

¹ <http://kickthejetengine.com/langorhythm/>

² See <https://youtu.be/ZhK0Uurb5Vw> for a video showcase.

Fragile Pulse: A Meditation App

Kyle Booten

Neukom Institute for Computational Science
Dartmouth College
Hanover, NH 03755 USA
kyle.p.booten@dartmouth.edu

Abstract

This paper presents *Fragile Pulse*, an (anti-)interactive work of electronic literature about the distracting nature of digital media. The calming text demands the reader's silence and stillness; otherwise the text itself becomes distracted and anxious.

Introduction

Contemporary digital media is reshaping the way our minds work, often in extremely dangerous ways. The internet—with its endless forking paths of links and constantly-updating feeds—is making our brains less content with and capable of linear attention (Carr 2011; Hayles 2007). It is both addictive (Young 1998) and a risk-factor for depression and anxiety (Woods and Scott 2016; Young and Rogers 1998). Still, a new wave of “self-care” and “mindfulness” apps have promised new ways of caring for one's mental state. Headspace, Shine, and Calm provide users with features such as short guided meditations, daily inspirational quotes, and soothing music to promote sleep. Another app, Forest, “game-ifies” the mere act of not using one's phone. Will our computers and smartphones save us from the problems they have caused?

“Fragile Pulse” is a work of electronic literature that confronts the viewer with this question by representing the tension between linear, “deep” attention and the skittering, impatient “hyper” attention encouraged by digital media (Hayles 2007).

Fragile Pulse

Fragile Pulse presents the reader with a calming, human-authored, linear text, not unlike those that abound on mindfulness apps. Line-by-line, it pulses on the screen roughly at the speed of a relaxed breath. However, this meditative unfolding of text can be disrupted. The program monitors the viewer with a web-camera and microphone. Any sound or movement above certain thresholds will initiate one or more “distractions,” computer-generated deviations from the core text. Contrasting the calming pulse of the main text, these distractions vibrate and blink, drawing attention to themselves in a way that evokes pop-up advertisements. The more sound and movement the system detects, the more dis-

tractions clutter the screen. To re-initiate the main, meditative text, the reader must return to a state of silence and stillness; gradually the distractions will dissipate, allowing the main text to pulse onward. However, sounds will also litter the screen with permanent “ear” emoji, movements with “eye” emoji—a way of keeping score.

Generating Distractions

The core of *Fragile Pulse* is a Natural Language Generation system. This system contains a series of functions that, given some input, produce a nonlinearly-related output, a textual representation of the distracted mind's lines of flight.

For instance, given the line “You are eating a perfect blade of grass,” the system may target a word within this line (a noun, verb, or adjective) and then leap to an orthographically similar word in a large corpus (e.g. “glass,” or “perfect”). However, to echo the way that distracted thoughts can quickly spiral into anxious ones, the system does not choose them randomly. Rather, according to a principle of *Affective Filtering*, it is more likely to choose those that are semantically close to a negative word (such as “terrible” or “pain”) according to a vector-space model of language (Mikolov et al. 2013).¹ The principle of Affective Filtering applies to other functions that generate distractions. One function, given a target word, will wander to a semantically-related word according to that vector-space model, again prioritizing words that are close to certain negative terms. Other functions rely on semantic relations mined from a large number of Project Gutenberg texts using SpaCy's dependency parser (Honnibal and Johnson 2015). Given a noun such as “wolf,” it will choose a transitive verb of which this noun is the subject according to (noun, verb) patterns mined from the corpus, prioritizing those verbs that are close to “kill” or “hate” (e.g. “bite”).² This produces an anxious question such as “What if a wolf bites me?” Using (adjective, noun) pairs mined from this

¹Cosine similarity between Google News vectors was used. See: <https://code.google.com/archive/p/word2vec/>

²Sometimes antonyms (e.g. “terrible” and “wonderful”) may be judged to be similar by this model; additional filtering omitted words that were closer to certain positive words than certain negative words. Likewise, semantic proximity between (noun, verb) and (adj, noun) pairs was used to emphasize related words.

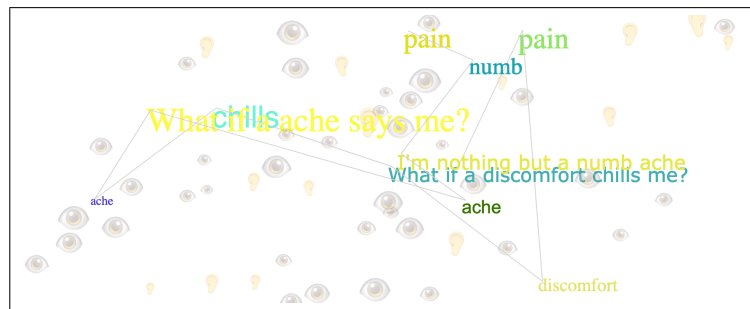


Figure 1: A fit/flight of distractions. Lines chart the semantic wandering. Eyes and ears document movement and sound.

corpus and prioritizing adjectives close to words like “terrible” and “useless,” another function produces self-critical statements like “I’m nothing but a stagnant lake.”

Other functions mimic anxious and distracted thinking while also adding linguistic variety. Using the Twitter API, for instance, one function returns a tweet that contains both the target word and one of a series of emoji that tend to signify sadness or anxiety. Another returns a statement of consumeristic desire extracted from Amazon product reviews (McAuley et al. 2015).

Chains of Distraction A key feature of the Natural Language Generation system is that the distraction functions can be arbitrarily chained together. When noise or movement disturbs the main text, the first distraction always takes as its point of departure whatever line of the main text was last pulsing on the screen. However, further distractions leap from the previous distraction, creating associative sequences meant to give the impression of a mind bouncing between thoughts as well as between moods. Disrupting the line “You are standing beneath a solemn moon...” may lead to a chain of distractions such as:

solemn
 sorrowful
 I am nothing but a sorrowful foreboding.
 foreboding
 dreading
 I am already dreading work next week.

Anti-Interactivity

Computer-generated literature often takes the form of algorithms that operate autonomously. For instance, Nick Montfort’s *Taroko Gorge* (2009) slowly and perhaps meditatively generates a poem line by line, whether or not the reader is following along. Other works in this field are interactive. For example, Camille Utterback and Romy Achituv’s *Text Rain* (1999) requires the reader to become a kind of dancer, catching letters with their limbs as they cascade down a screen. This piece, *Fragile Pulse*, offers a different sort of relation between the reader and the text, a text that only becomes legible with some guarantee of the reader’s attention. Physical stillness and silence have traditionally facilitated

certain forms of highly-attentive reading as well as meditation. In this case, technologies of interactivity (e.g. motion sensing) are deployed only to goad the reader toward the kind of mindful consumption most threatened by digital media. It remains to the reader to decide whether to follow this direction or to “fail” (i.e. never progress through the main, meditative text). Indeed, they may decide that it is more fun to distract the system. But a choice must be made.

References

- Carr, N. 2011. *The shallows: What the Internet is doing to our brains*. WW Norton & Company.
- Hayles, N. K. 2007. Hyper and deep attention: The generational divide in cognitive modes. *Profession* 2007(1):187–199.
- Honnibal, M., and Johnson, M. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 1373–1378. Lisbon, Portugal: Association for Computational Linguistics.
- McAuley, J.; Targett, C.; Shi, Q.; and Van Den Hengel, A. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 43–52. ACM.
- Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- Montfort, N. 2009. *Taroko gorge*. nickm.com/taroko_gorge/.
- Utterback, C., and Achituv, R. 1999. Text rain. *SIGGRAPH Electronic Art and Animation Catalog* 78.
- Woods, H. C., and Scott, H. 2016. #sleepyteens: Social media use in adolescence is associated with poor sleep quality, anxiety, depression and low self-esteem. *Journal of adolescence* 51:41–49.
- Young, K. S., and Rogers, R. C. 1998. The relationship between depression and internet addiction. *Cyberpsychology & behavior* 1(1):25–28.
- Young, K. S. 1998. Internet addiction: The emergence of a new clinical disorder. *Cyberpsychology & behavior* 1(3):237–244.

Emotion-driven Creative Music Composition in brAInMelody®

Masayuki Numao

The Institute of Scientific and Industrial Research, Osaka University
Osaka, Japan
numao@sanken.osaka-u.ac.jp

Abstract

The purpose of music creation is to cause impression to listeners. Based on this way of thinking, while watching the reaction of the listener, we will describe the artificial intelligence to perform creation. In this method, based on the reaction of the listener, that is, the measurement result of the emotion, AI predicts the feeling to the new song being made. In this way, having discussed automatic composition, we discuss AI approaches to activation of a human being (brain) by songs. This system demonstrates our emotion-driven creative music composition, which composes a piece of music on the fly based on physiological sensing, such as brain waves, heart rate and skin conductance, of the user.

Introduction

What is considered as a frontier of artificial intelligence is creative acts, such as novels, music, or the art of hospitality. Although authoring novel by artificial intelligence and automatic composition by Google are hot topics, people have to give their seeds of creation. In this extended abstract, from the viewpoint of entertaining people, we discuss on the flexibility in artificial intelligence to raise autonomy. That is, instead of the seeds of creation that a person gives to artificial intelligence, we detect the feelings of recipient to reduce the given seeds.

The purpose of creation is to inspire a recipient (in the case of music, a listener). To make it happen based on this concept, listener's reaction is detected while listening the created contents. Reaction of the listener, i.e., feelings is measured. In addition, the system predicts the response of the listener in order to avoid making queries many times.

Emotional measurement techniques targeting music is discussed for prediction technology and automatic composition based on them. By listening to music, we are in a healthy and comfortable state. It is trying to automate the music selection for that in music recommendation systems. Our aim is to extend it to implement automatic composition for this purpose.

Music is a flow of information among its composer, player and audience. A composer writes a score that players play to create a sound to be listened by its audience. Since a score, a performance or MIDI data denotes a section of the flow; we can know a feeling caused by a piece of score or

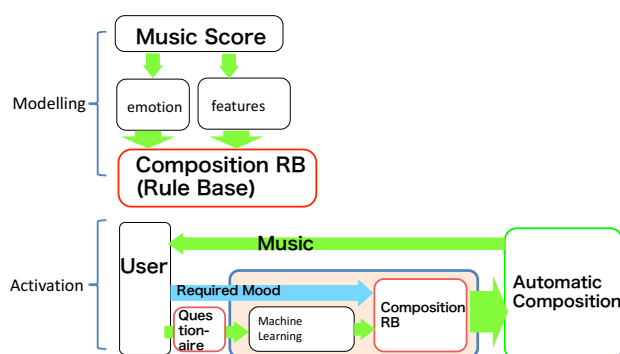


Figure 1: Contents creation based on feelings

performance. A feeling consists of very complex elements, which depend on each person, and are affected by a historical situation. Therefore, rather than clarifying what a human feeling is, we would like to clarify only musical structures that cause a specific feeling. Based on such structures, the authors constructed an automatic arrangement and composition system producing a piece causing a specified feeling on a person.

The system first collects feelings of a person for some pieces, based on which it extracts a common musical structure causing a specific feeling. It arranges an existing song or composes a new piece to fit such a structure causing a specified feeling. Figure 1 is a diagram for the creation based on queries to the user.

It has three levels of composition process: (a) chord-progression and motif level (Figure 2)(Numao, Takagi, and Nakamura 2002; Otani, Kurihara, and Numao 2012), (b) rhythm level (Otani et al. 2013), and (c) melody level (Otani, Okabe, and Numao 2018).

Figure 3 is a diagram for the creation based on physiological sensors.

Demonstration

As such, we address the problem of determining the extent by which emotion-inducing music can be modeled and generated using creative music compositional AI. Our approach involves inducing an affects-music relations model that describes musical events related to the listener's affec-

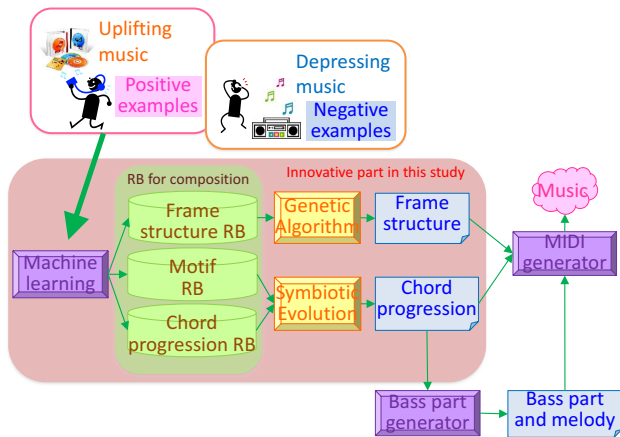


Figure 2: Music-composition flow at the chord-progression and motif level



Figure 4: The brain-wave headset with a headphone

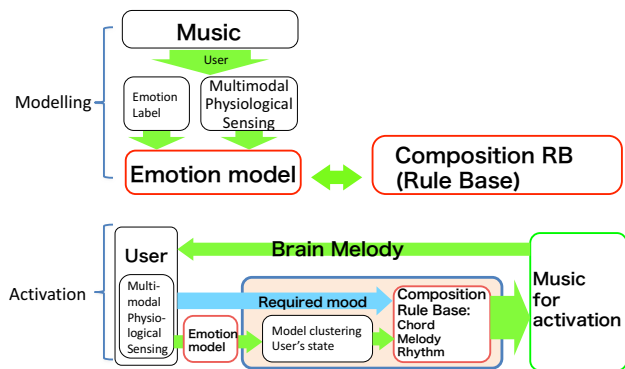


Figure 3: Sensor-based creation

tive reactions and then using the predictive knowledge and character of the model to automatically control the music generation task. We have embodied our solution in a Constructive Adaptive User Interface (CAUI) that re-arranges or composes a musical piece based on one's affect.

This demonstration shows brAInMelody® system constructed based on these series of research(Numao, Takagi, and Nakamura 2002; Numao, Kobayashi, and Sakaniwa 1997; Numao, Takagi, and Nakamura 2002; Sugimoto et al. 2008; Otani, Okabe, and Numao 2018; Otani et al. 2013; Otani, Kurihara, and Numao 2012; Cabredo et al. 2013; Thammasan et al. 2016; 2017b; 2017a).

References

Cabredo, R.; Legaspi, R.; Inventado, P. S.; and Numao, M. 2013. Discovering emotion-inducing music features using EEG signals. *Journal of Advanced Computational Intelligence and Intelligent Informatics* 17(3):362–370.

Numao, M.; Kobayashi, M.; and Sakaniwa, K. 1997. Acquisition of human feelings in music arrangement. In *Proc. IJCAI 97*, 268–273. Morgan Kaufmann.

Numao, M.; Takagi, S.; and Nakamura, K. 2002. Construc-

tive adaptive user interfaces — composing music based on human feelings. In *Proc. Eighteenth National Conference on Artificial Intelligence (AAAI-02)*, 193–198. The AAAI Press / The MIT Press.

Otani, N.; Kamimura, R.; Yamano, Y.; and Numao, M. 2013. Generation of rhythm for melody in a constructive adaptive user interface. In *4th International Workshop on Empathic Computing, a workshop in IJCAI-13*.

Otani, N.; Kurihara, S.; and Numao, M. 2012. Generation of chord progression using harmony search algorithm for a constructive adaptive user interface. In *Proc. 12th Pacific Rim International Conference on Artificial Intelligence (PRICAI 2012), Lecture Notes in Artificial Intelligence*, volume 7458, 400–410.

Otani, N.; Okabe, D.; and Numao, M. 2018. Generating a melody based on symbiotic evolution for musicians' creative activities. In *Proc. Genetic and Evolutionary Computation Conference (GECCO'18)*, 197–204.

Sugimoto, T.; Legaspi, R.; Ota, A.; Moriyama, K.; Kurihara, S.; and Numao, M. 2008. Modelling affective-based music compositional intelligence with the aid of ANS analyses. *Knowledge-Based Systems* 21(3):200–208.

Thammasan, N.; Moriyama, K.; Fukui, K.; and Numao, M. 2016. Continuous music-emotion recognition based on electroencephalogram. *IEICE Transactions* E99-D(4):1234–1241.

Thammasan, N.; Hagad, J. L.; Fukui, K.; and Numao, M. 2017a. Multimodal stability-sensitive emotion recognition based on brainwave and physiological signals. In *2017 Seventh International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW), IEEE Xplore*, 44–49.

Thammasan, N.; Moriyama, K.; Fukui, K.; and Numao, M. 2017b. Familiarity effects in EEG-based emotion recognition. *Brain Informatics* 4:39–50.

Creative Sketching Partner: A Co-Creative Sketching Tool to Inspire Design Creativity

Nicholas Davis¹, Safat Siddiqui¹, Pegah Karimi¹, Mary Lou Maher¹, Kazjon Grace²

¹UNC Charlotte, ²The University of Sydney

¹USA, ²Australia

ndavis64@uncc.edu, ssiddiq6@uncc.edu, pkarimi@uncc.edu, m.maher@uncc.edu, kazjon.grace@sydney.edu.au

Abstract

The Creative Sketching Partner is an AI-based co-creative sketching tool that supports the conceptual design process. This AI partner presents sketches of varying visual and conceptual similarity based on the designer's sketch. The goal of the partner is to present a sketch to inspire the user to explore more of the design space and to reduce design fixation, i.e. becoming stuck on one or a class of designs during the design process. The system is meant to help designers achieve a conceptual shift during their design process by presenting similar designs or images from different domains. Users can control the parameters of the algorithm by specifying how visually and conceptually similar the system's sketch should be to their own.

The Creative Sketching Partner

Sketching is a critically important component of design creativity that facilitates thinking, reflection, and enables designers to share their ideas with other stakeholders. Designers often rapidly iterate through their initial sketches in the early stages of design, allowing them to ideate and explore the conceptual space of the design. However, designers sometimes face design fixation (Purcell and Gero 1996), or becoming stuck on one design or a class of designs. To inspire design creativity and overcome design fixation, we have developed the Creative Sketching Partner (CSP).

The Creative Sketching Partner is a co-creative sketching tool that analyzes the user's sketch and produces a sketch from a large database of sketches that has some visual and conceptual similarity with the user's sketch. The goals is that the CSP will enable designers to experience conceptual shifts in the design process by presenting sketches that bear some visual and conceptual resemblance to the initial sketch. Analogical reasoning may be triggered by exposure to a conceptual shift stimulus. Analogy is a common activity in design, in which a source object is identified that can be mapped onto the current design (the "target"), and then some properties of that source can be transferred to the target to help solve the design problem (Grace, Gero, and Saunders 2015).

Co-creative sketching systems are an active area of research in the computational creativity community. One such example is the Drawing Apprentice, which is a co-creative

drawing partner that collaborates with users in real time (Davis et al. 2015). The system uses sketch recognition to identify objects drawn by the user and selects a complementary object to display on the screen. Instead of selecting a sketch from the same conceptual category, such as Drawing Apprentice, the CSP uses a computational model of conceptual shifts (Karimi et al. 2018) to determine an appropriate target sketch from a dataset.

Conceptual Shift Algorithm

The AI model for determining conceptual shifts has two components: visual similarity and conceptual similarity. Visual similarity entails identifying a sketch that shares some structural characteristics, whereas conceptual similarity identifies a concept that has some semantic relationship.

The visual similarity module computes the distances between the cluster centroids of distinct categories and maps the user's input to the most similar sketches from other categories. First, a deep learning model is employed called Convolutional Neural Network-Long Short Term Memory (CNN-LSTM) (Carbune 2017). We train the model from scratch on the QuickDraw! dataset of 345 categories of human-drawn sketches (Jongejan et al. 2016). Each sketch is represented by the last LSTM layer, for 256 values per sketch. We use the resulting feature vectors for sketches in each category to create clusters of visually similar sketches. This process provides a feature vector representation for calculating the novelty between the user's initial sketch and sketches in the QD dataset using visual similarity.

The conceptual similarity module takes the pairs of selected category names from the previous step and computes their semantic similarity. This module uses a word embedding model (Mikolov 2016) trained on the Google News corpus with 3 million distinct words. The visual similarity module provides a set of candidate sketches to the conceptual similarity module. We extract the word2vec word embedding features (Mikolov 2016) from these category names. The similarity between the category of the source sketch and the selected target sketch is computed as $1 - d_c$, where d_c is the cosine distance between the feature vectors of category names. The larger number indicates that the two sketch categories are more likely to appear in the same context, whereas a smaller number indicates that the two are less associated with each other.

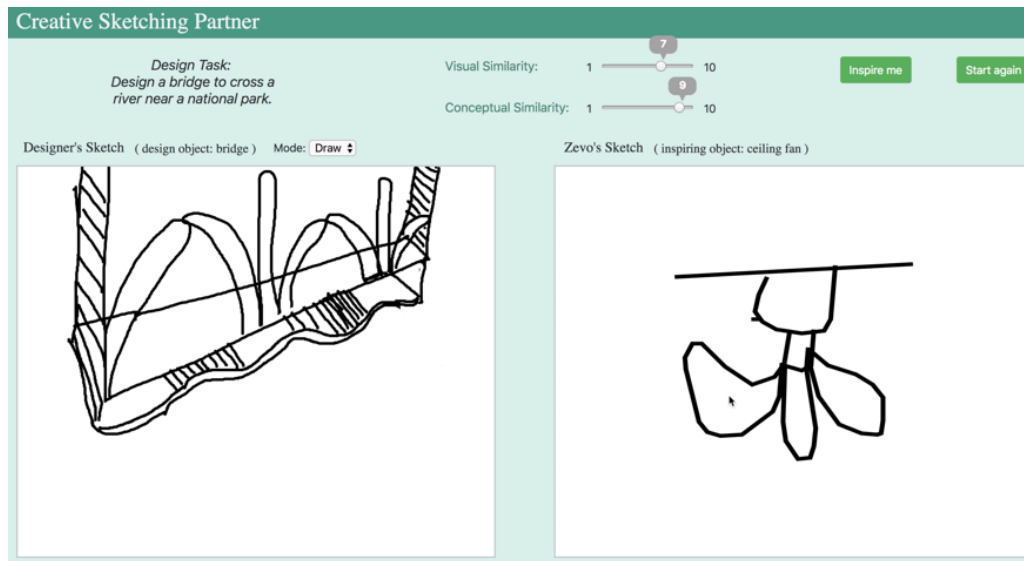


Figure 1: The Creative Sketching Partner interface and example sketch from a user.

User Experience

The Creative Sketching Partner is implemented as a web-based co-creative partner to assist in the process of sketching concepts during the early stage of the design process. In this tool, the user is provided with a prompt to draw a particular type of object, such as a chair, bridge, or car. In addition to the prompt, the user is provided with a context to help situate the design in a scenario, such as 'a bridge to cross a river near a national park.'

The left drawing panel is the users and provides basic sketch functionalities, such as drawing in black ink and erasing that ink. The right drawing panel shows the agents (named Zevo) sketch. Above each panel is a label showing what the object in the panel is supposed to be. The label can be useful to interpret the agents drawing as it sometimes selects a somewhat abstract representation of the concept to display.

Once the user sketches their design, they can activate the agent by clicking the Inspire me button in the top-right of the interface. Users can customize the agents procedure by specifying how similar they would like the agents sketch to be to their own. Both the visual and conceptual similarity can be selected along a scale of 1 to 10. The system uses the selected parameters and the users sketch to determine the most appropriate sketch response from its database of sketches to display on the interface. After each response, the user can iterate on their designs and then explore new system sketches based on their current design. The user can also erase their design and start over.

We have studied the use of the Creative Sketching Partner with about 50 participants. Overall, the participants found that sketching with the CSP lead them to various ways of changing their own design from simple combinatorial processes to more transformative designs. Users incorporated elements from the system's sketch into their own to add fea-

tures, functions, and context to their design sketch. The object label as well as the visual features of the object were utilized to inform the user's design. In some instances, the participants inferred functions of the inspirational object that were not included in sketch but derived from the features of the semantic category of the object.

References

- Carbone, V. 2017. Recurrent neural networks for drawing classification. https://www.tensorflow.org/tutorials/sequences/recurrent_quickdraw.
- Davis, N. M.; Hsiao, C.-P.; Singh, K. Y.; Li, L.; Moningi, S.; and Magerko, B. 2015. Drawing apprentice: An enactive co-creative agent for artistic collaboration. In *Creativity & Cognition*, 185–186.
- Grace, K.; Gero, J.; and Saunders, R. 2015. Interpretation-driven mapping: A framework for conducting search and rerepresentation in parallel for computational analogy in design. *AI EDAM* 29(2):185–201.
- Jongejan, J.; Rowley, H.; Kawashima, T.; Kim, J.; and Fox-Gieg, N. 2016. The quick, draw!-ai experiment. *Mount View, CA, accessed Feb 17:2018*.
- Karimi, P.; Grace, K.; Davis, N.; and Maher, M. L. 2018. Creative sketching apprentice: Supporting conceptual shifts in sketch ideation. In *International Conference on Design Computing and Cognition*, 721–738. Springer.
- Mikolov, T. 2016. word2vec: Tool for computing continuous distributed representations of words. <https://code.google.com/p/word2vec>.
- Purcell, A. T., and Gero, J. S. 1996. Design and other types of fixation. *Design studies* 17(4):363–383.

Two Fragrances Designed for Brazilian Millennials

¹David Apel, ¹Sophie Bensamou, ¹Veronique Ferval, ²Jing Fu, ²Richard Goodwin,
²Christian Harris, ¹Andrea Lombardi, ²Robin Lougee,
²Joana Maria, ²Richard Segal, ²Tenzin Yeshi
¹Symrise, New York City, NY 21031 USA
²IBM Research, Yorktown Heights, NY 10598 USA
rgoodwin@us.ibm.com

Abstract

There is an art and a craft to making an appealing fine fragrance that takes perfumers many years to master. The *IBM Research AI for Product Composition* system was created to assist perfumers to design novel fragrances that are appealing, well balanced and technically sound. The system has been used by Master Perfumers at Symrise to develop 2 fragrances for Brazilian millennials. This creative submission presents the two fragrances, one designed for men and the other for women.

Introduction

Symrise AG and IBM Research are exploring the use of computational creativity to aid professional perfumers in the design of new fragrances. Our collaboration has produced the first AI-enabled commercial fine fragrances. These fragrances were developed for the millennial market in Brazil and will be on sale in May 2019 (Goodwin 2019).

Perfumery is both a craft and an art. A good fragrance needs to be well balanced, with top, middle and base notes that complement each other. It needs to last on the skin and smell good both when first applied and 8 hours later when many of the more volatile elements have evaporated. It needs to be shelf-stable for at least 3 months and not separate or form a precipitate. The fragrance must be safe to apply to the skin. It can't be a skin irritant and it needs to meet a large and ever-changing set of regulatory requirements. Perfumers learn the craft of perfumery in perfumery school and through apprenticing to a master perfumer. To become fully trained takes a perfumer 7 to 10 years.

Beyond the craft of creating a good fragrance, the perfumer also needs to be an artist designing unique fragrances that appeal to the nose and the emotions. Smell is one of the most primitive of our senses and linked closely with memory and emotion (Herz 2004). The right fragrance can help convey the personality of the wearer and create the right emotional environment.

To design a perfume, a perfumer iteratively creates trial formulas, has them compounded by a robot and smells them. They typically work with an evaluator who also smells the



Figure 1: Image of the two fragrances.

sample and provides feedback. The process continues until the perfumer and evaluator have arrived at a sample or two that they want to send to the client. The client may have requested samples from multiple fragrance houses and will select a winner. The client may request changes and are likely to put the selected fragrances into a consumer test with people from the target market segment before putting the fragrance on the market.

The Fragrances

IBM Research AI for Product Composition is a co-creative system that applies computational creativity to fragrance design. The system takes one or more inspiration fragrances to indicate the general area of the fragrance space to explore. The system also takes a creativity level from 1 to 10 as input. Specifying a large creativity level causes the system to explore a very wide area of the fragrance space. Specifying a lower creativity level causes the system to focus on the region of the inspiration set more closely and fine tune the fragrances.

The first fragrance we present is designed for millennial women in Brazil. The fragrance is a well-balanced floral fragrance that is fresh and modern. The main structure is similar to a classic floral violet with white flower, raspberry and a passion flower accords.

The formula generated by the system was adjusted slightly by the master perfumer to increase its freshness. One ingredient was decreased from 1.35% of the formula to 1.0% of the formula, a total change of 0.35% from the generated formula. The result has performed very well in consumer tests with millennial women in Brazil.

The women's fragrance formula was interesting not only because it has an pleasant scent, but also the way it achieved the result. The formula used a large amount of an expensive Chinese floral absolute. Typically, a perfumer would not use such an expensive ingredient in large amounts in a mass-market fine fragrance because it makes it hard to achieve the target raw material cost. In this case, the system was able to balance the expensive ingredient with a set of less expensive ingredients to create a beautiful fragrance that met the cost target. This is an example of how the system can spur creativity by suggesting ways of achieving a goal that the perfumer would not normally consider.

The millennial men's fragrance has the structure of a classic fougère, a common style for men's fragrances. It goes beyond the classic style to add gourmand elements of cardamom, fenugreek, tonka lactone and a lait chaud (hot milk). The combination of elements makes the fragrance unique and the balance between the various elements makes the fragrance appealing to the target market, millennial men in Brazil. The master perfumer did make one adjustment to the formula, reducing the tonka lactone by 0.04%

Fragrance Creation System

IBM Research AI for Product Composition, sketched in figure 2, is a data driven machine learning system that helps the perfumer iterate over designs by suggesting alternative formulations for olfactory experiences under development. The system makes suggestions by learning from Symrise's extensive historical repository of formulas and sensory tests.

The formula generator uses a generate and test methodology in which multiple candidate formulas are created and then evaluated. The generation process is designed to mimic how perfumers usually learn and work. The training data for the system consists of 100,000s of previously created formulas. The data includes not only the final formula, but all the intermediate formulas created en-route to the final formula. The system learns about raw material substitutes, complements and dosing. It learns about formula structure and how the structure for a fine fragrance differs from the structure for a shampoo or a candle fragrance. The system also learns a fragrance space distance and a model to predict likely success.

Since novelty is key to designing a new fragrance, the fragrance distance we learn is critical for the system to identify novel fragrances. There are many ways to achieve the same olfactory result and two formulas that share no ingredients may be indistinguishable to most people. On the other hand, two formulas that share all the same ingredients but have different dosing may smell completely different. Our distance model tries to predict which fragrance formulas people will judge to be more similar. This allows the system

to identify when a fragrance formula is not too similar to a formula already on the market.

Once the system has generated formulas, identified which are valid for the intended application and removed those that are too similar to existing formulas, we score them on likely success in the intended market. For each historical fragrance we know the intended market, which formulas were selected to send as samples, whether the sample won with the client, how well it did in consumer tests and its sales longevity. A variety of fragrance formulas with the highest scores are presented to the master perfumer for consideration.

IBM Research AI for Product Design

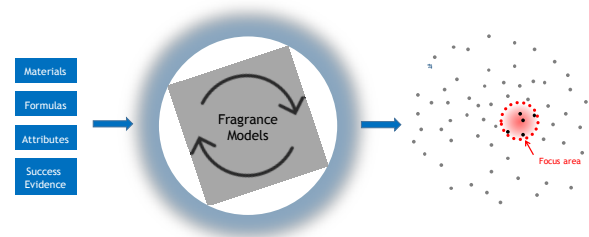


Figure 2: IBM Research AI for Product Design learns to produce fragrances based on historical product data, an inspiration set and a desired level of creativity.

Conclusion

The first fragrance products developed with *IBM Research AI for Product Composition* will be available for retail sale in Brazil from O Boticario in May 2019.

The underlying technology can be generalized to other products such as cosmetics, flavors (Lougee 2019), detergents, adhesives, lubricants and construction materials.

Acknowledgements

We acknowledge our colleagues, Christian Schepers and Saurin Patel for their contributions to this work.

References

- Goodwin, R.; Maria, J.; Das, P.; Horesh, R.; Segal, R.; Fu, J.; and Harris, C. 2017. AI for Fragrance Design. In *NIPS Workshop on Machine Learning for Creativity Design*.
- Lapid, H.; Harel, D.; and Sobel, N. 2008. Prediction models for the pleasantness of binary mixtures in olfaction. *Chemical senses*, 33(7), 599-609.
- Goodwin, R. 2019. Using AI to Create New Fragrances, <https://www.ibm.com/blogs/research/2018/10/ai-fragrances/>.
- Herz, R.S.; Eliassen, J.; Beland, S.; and Souza, T. 2004. Neuroimaging evidence for the emotional potency of odor-evoked memory. *Neuropsychologia* 42 (2004), 371-378

Computational-Creativity Enabled One Pan Seasonings for Retail Sale

¹Jing Fu, ¹Richard Goodwin, ¹Christian Harris, ¹Kimberly Lang, ¹Robin Lougee, ²CJ McLane,
¹Joana Maria, ²Jim Martin, ¹Richard Segal, ¹Tenzin Yeshi

¹IBM Research, Yorktown Heights, NY 10598 USA
²McCormick & Company, Hunt Valley, MD 21031 USA
rlougee@us.ibm.com

Abstract

Everyone eats. Many people cook. But few are aware of the unique creative challenges facing professional product developers in flavor and food product companies. The *IBM Research AI for Product Composition* is a system to assist product developers with these real-life creativity challenges for which no AI-solution existed. The system for compositional product development is in use at McCormick & Company, a global leader in flavor. We present the first set of products targeted for retail sale that has been enabled by the system. The three seasoning mixes for one dish meals are planned to be on retail shelves mid-2019.

Introduction

McCormick & Company and IBM Research are pioneering the use of computational creativity to aid professional flavor and food product developers. Our multi-year collaboration has produced the first AI-enabled retail products that will be available on grocer's shelves in 2019 (McCormick 2019, Lougee 2019).

Work in computational creativity typically approaches the culinary arts from the perspective of a chef creating a dish represented by recipe (Pinel *et al.* 2015, Anorim *et al.* 2017). We address a different scenario: a professional product developer creating a commercial product represented by a formula. The different objectives, constraints, assumptions and data available for product development require new approaches for creative generation and evaluation.

A product developer may have a target flavor experience objective (e.g., "Tuscan Chicken"). In such a case, the challenge is to create a product formula that will produce an outstanding flavor experience in the final dish prepared by the home cook. It should be highly and consistently liked across the target consumer segment. The product developer must consider a host of constraints (e.g., natural, Halal, nutrition, manufacturability, shelf life, country regulations) which affect the choice and amount of ingredients in the formula.

A product developer iteratively creates trial formulas, compounds samples, and runs a variety of taste and possibly consumer tests on the samples. Feedback on the samples allows the developer to learn what works and suggests how to

improve the formula. It is an art and a science. A junior product developer may take three times as many iterations as a senior product developer to create a ready-to-commercialize formula.



Figure 1: Image of a dish being cooked using a new seasoning mix co-designed using computational creativity.

The Seasonings

IBM Research AI for Product Composition is a co-creative system in use at McCormick & Company that applies computational creativity to real-life product development. No such similar system exists today as far as we know. The system incorporates several new technologies including the use of flavor distance for the generation of candidate formulas; user-controlled creativity levels; the ability to learn to predict formula success using direct (e.g., taste tests) and indirect (e.g., sales) success measures. The system does not depend on the flavor-pairing hypothesis (Ahn *et al.*, 2011) which makes it applicable globally as the flavor-pairing hypothesis is believed not to hold in some cultures. The three flavors that we will present at the conference were developed by product developers at McCormick. The flavors are part of McCormick's new ONE product line which provides seasoning mixes for single pan dishes. The three flavors are "Tuscan Chicken & Vegetables," "Bourbon Pork Tenderloin & Vegetables," and "New Orleans Style Sausage & Vegetables." The product developers who created these dishes report that the AI system made several novel

suggestions that were ultimately included in the final product and helped the products perform well in a consumer panel. These are the first food products developed using computational creativity that are available for retail sale.

Flavor Creation System

IBM Research AI for Product Composition is a data driven machine learning system that helps the developer iterate over designs by suggesting alternative formulations for flavor experiences under development. The system makes suggestions by learning from McCormick's extensive historical repository of formulas and taste tests.

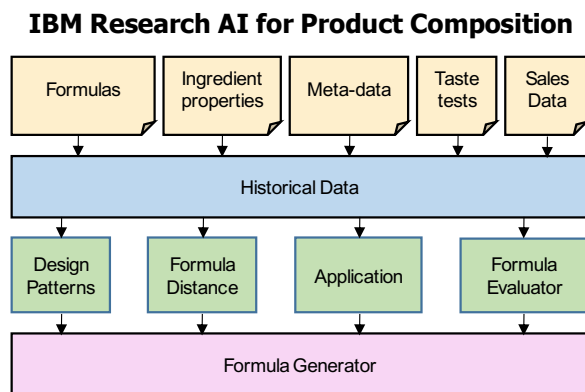


Figure 2: *IBM Research AI for Product Composition* learns to produce product formulations based on historical product data, ingredient properties, product meta-data, the results of taste tests, and sales information.

The formula generator uses a generate and test methodology in which candidate formulas are created and then evaluated using a success model. The generation process is designed to mimic how product developers usually work. The training data for the system consists of 100,000s of previously created formulas. The data includes not just the final output of the creative process, but all the intermediate product formulations created during a product's development. It mines this data to identify the design patterns that developers typically employ when perfecting a formula. The candidate generation algorithm uses the learned design patterns to search the space of possible formula improvements in a manner similar to how a product developer would search.

The formula evaluator analyzes each proposed candidate and assigns a score that is indicative of how well a formulation is likely to do in the market. Flavor is a very complex human experience. Theoretical constructs such as the flavor pairing hypothesis, olfactory pleasantness (Lapid, Harel, and Sobel, 2008), and Bayesian surprise (Varshney, 2013) try to estimate some dimensions of this experience. But they are approximations. The best indication of how a formula will do in the market is human taste tests and consumer panels. Instead of trying to model human taste, we predict market success based on historical data. The primary source of data for learning the success model is consumer panels and

employee taste tests. We also learn from implicit indicators of success, such as sales longevity, to put a greater emphasis on successful formulations.

Conclusion

The first flavor products developed with *IBM Research AI for Product Composition* will be available for retail sale with the new ONE product family launching from McCormick in mid-2019. The computational creativity system helped the ONE product developers explore new creative ideas, accelerating the time to market while achieving high ratings with consumer testers.

Based on the promising results to date, McCormick plans to roll out the system globally to operations in more than 20 labs in 14 countries encompassing over 500 product and flavor developers and their support staff. The underlying technology can be generalized to other products such as cosmetics, fragrances (Goodwin 2017), detergents, adhesives, lubricants and construction materials.

Acknowledgements

We would like to acknowledge our McCormick & Company colleagues, Alain Chartrand, Terry Moon, Karyn Ruocco, Joel Samuel, and Hennie Van Zuylen for their valuable contributions to this work.

References

- Ahn, Y.; Ahnert, S.; Bagrow J.; and Barabasi, A. 2011. Flavor network and the principles of food pairing. *Scientific Reports*, 1:196–202, December 2011.
- Amorim, A.; Góes, L.F.; Silva, A.R.; and França, C. 2017. Creative flavor pairing: using RDC metric to generate and assess ingredient combination. *Proc. 8th Int. Conf. on Computational Creativity (ICCC 2017)*, 33–40.
- Goodwin, R.; Maria, J.; Das, P.; Horesh, R.; Segal, R.; Fu, J.; and Harris, C. 2017. AI for fragrance design. *NIPS Workshop on Machine Learning for Creativity Design*.
- Lapid, H.; Harel, D.; and Sobel, N. 2008. Prediction models for the pleasantness of binary mixtures in olfaction. *Chemical senses*, 33(7), 599-609.
- Lougee, R. 2019. Using AI to create new flavor experiences. <https://www.ibm.com/blogs/research/2019/02/ai-new-flavor-experiences/>.
- McCormick & Company, 2019. McCormick & Company and IBM announce collaboration pioneering the use of artificial intelligence in flavor and food product development. <https://mccormickcorporation.gcs-web.com/news-releases/news-release-details/mccormick-company-and-ibm-announce-collaboration-pioneering-use>.
- Pinel, F.; Varshney, L.R.; and Bhattacharjya, D. 2015. A culinary computational creativity system. In *Computational Creativity Research: Towards Creative Machines*, T. R. Besold, M. Schorlemmer, and A. Smaill (eds.), Springer.
- Varshney, L.R. 2013. To surprise and inform. *IEEE International Symposium on Information Theory*.

ChordAL: A Chord-Based Approach for Music Generation using Bi-LSTMs

Tan Hao Hao

School of Computer Science and Engineering
Nanyang Technological University, Singapore
tanh0207@e.ntu.edu.sg

Abstract

In this paper we propose ChordAL, a chord-based generation system which is capable of composing melodies based on a few starting chords. It consists of 3 components: a *chord generator*, a *chord-to-note generator*, and a *music styler*. In our chord generator, our learnt chord embeddings surprisingly unveil the Circle of Fifths, which shows that our model is able to learn hidden musical structure through a simple chord generation task. The chord-to-note generation is treated as a *seq2seq* task like machine translation, and we make use of the chord embeddings learnt previously for melody generation. Evaluation results show that ChordAL performs well in generating fluent harmony due to its harmony-based nature, although its performance on rhythm and structure needs further improvement.

Introduction

Algorithmic composition of music using deep learning has gained increasing attention in recent years, with a large number of proposed music generation systems in the field. However, not much work has been done on analyzing *chord-based approaches* - particularly, how to compose melodies based on a given chord progression. Yet, it seems to be a very intuitive approach for human composers to write songs based on chord progressions, especially in Western music genres like pop and jazz music.

Here we propose ChordAL, a chord-based generation system which is capable of composing melodies based on a few starting chords. The motivation is to investigate the capability of chord-based approaches in “translating” chord progressions into fluent, harmonic melodies.

Overview

ChordAL’s pipeline can be divided into 3-step:

- **Chord Generator:** First, a chord progression is generated from scratch given a few chords as starting seed.
- **Chord-to-Note Generator:** Next, the sequence of chords is fed into the Chord-to-Note Generator, that generates the melody line based on the given chords.
- **Music Styler:** Finally, both the generated chord and melody are fed into the Music Styler component for post-

processing and styling, in order to combine both parts into a presentable piece.

Chord Generator

Chords are first encoded as 24 different indices, which represents 12 different pitches in an octave with *major* and *minor* chord each. During preprocessing, we decide to remove the repeated chords to ensure that all adjacent chords are different. Generation results show that this greatly helps to increase the variation and quality of the generated chord progressions.

For the model, we pass the chord vectors into a 32-dimension *embedding layer*. The embeddings are then fed into a stacked Bi-LSTM of 2 layers, with 64 hidden neurons each. A dense layer is added for the output with a softmax activation function for multi-class classification to predict the output chord index.

The concept of using *embeddings* is borrowed from the natural language processing domain. Word embeddings are expected to contain meaningful representation of the words in the vocabulary, such as the *word2vec* model. Here, we hope that the embedding layer could learn meaningful representations about the relationships between chords, which will be useful for the chord-to-note generation later.

Chord-to-Note Generator

We use *piano-roll* representation for the melody, which can be interpreted as a matrix of shape $(128, t)$, where t is the length of the piano-roll.

For the model, the chord indices are converted to their corresponding embeddings learnt in Chord Generator, and then fed into a stacked Bi-LSTM of 2 layers with 64 hidden neurons each. Dropout layers of probability 0.2 are added after each Bi-LSTM layer. A *tanh* activation is added after the first dropout layer. A time-distributed dense layer is added for the output, with a softmax activation function for multi-class classification to predict the output note.

Music Styler

The Music Styler implements additional tuning on the generated melody line, such as removing random short notes to improve the fluency of the song. Then, it applies styling as defined by the user for both the chord and melody parts. We find that *sustaining instruments* (strings, organ, brass, etc.)

sound best on the compositions, as they are mainly chordal in nature.

Dataset

The following parallel chord-to-note datasets are used:

- **Nottingham dataset:** We use the cleaned version pre-processed by Jukedeck, with separated chord and melody parts for each piece.
- **McGill-Billboard Chord Annotations:** It contains chord annotations for around 1000 Billboard chart songs.
- **CSV leadsheet database:** A leadsheet database that contains around 2200 Western music pieces across different genres including rock, pop, jazz, etc.

All entries are indexed into a *chord database* and a *melody database*. We use the chord database to train the Chord Generator, and both databases to train the Chord-to-Note Generator. Each entry is transposed to all 12 different pitches in the octave for normalization.

Results and Discussion

Resemblance of major Circle of Fifths We extract the chord embeddings for each chord index and visualize them after applying Principal Component Analysis (PCA) to reduce the 32 dimensions into 2. As seen in Figure 1 below, the chords almost form the exact same circle as the major Circle of Fifths. This shows that our embedding layer is capable to learn the underlying structure of music, even with a simple chord generation task.

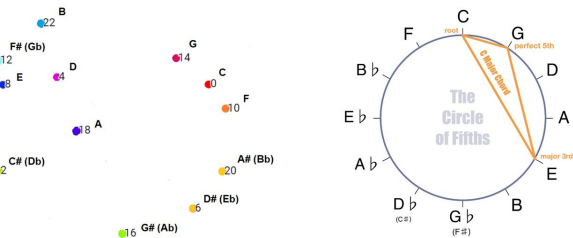


Figure 1: The learnt chord embeddings (left) is found to highly resemble the major Circle of Fifths (right).

Evaluating generated melodies Overall, the pieces generated by ChordAL sound pleasant and harmonic. This is accredited to the harmony-based nature of the approach itself, as the network is aimed to learn about the relationship between the notes and the chords. However in general, each melody note sustains for a long duration. One reason may be that the note representation in the piano roll is highly repetitive - sustained notes are encoded as the same one-hot vector across a long duration. This may cause our network to be biased against generating highly repetitive values that result in long notes.

Subjective Evaluation A comprehensive subjective evaluation is conducted on 5 of the songs generated by ChordAL, which prompts the respondents to rate its performance on a 5-point Likert scale based on harmony, rhythm and structure. A total of 15 respondents with a higher music proficiency level, which include professional piano players, band performers and music teachers, took part in this comprehensive evaluation. Figure 2 below shows the results of the evaluation.

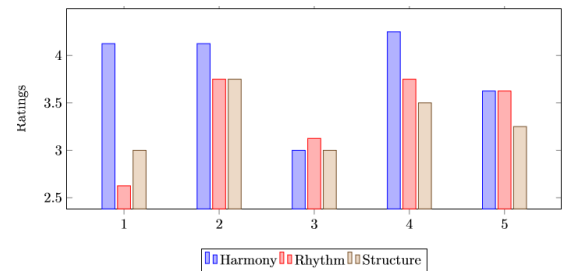


Figure 2: Results for subjective evaluation.

In general, ChordAL's compositions score highest in terms of harmony, with an average rating of 3.825. This further shows that a chord-based generation framework can guarantee harmonic, pleasant-sounding generation. However, ChordAL's compositions lack rhythm and structure, with an average rating of 3.375 and 3.3 respectively. We think that it is possible to build additional components on top of the chord-based framework that handle the aspects of rhythm and structure, while preserving the strength of this framework in composing fluent harmony.

Conclusion and Future Work

We propose a 3-step chord-based framework for melody generation, which is able to ensure fluent harmony in the generated pieces. We also show that chord embeddings could be learnt by a simple chord generation task, which they highly resemble the Circle of Fifths.

ChordAL is most suitable for generating chamber music and strings/woodwind ensemble pieces, as these types of music are chordal in nature. Future work will be focusing on increasing the vocabulary of chords used, as well as improving on the rhythm and structure in the compositions.

ChordAL is open-source and welcomes contribution at: <https://github.com/gudgud96/ChordAL>. All compositions by ChordAL can also be heard on Soundcloud at: <https://bit.ly/2HzighM>.

Acknowledgement

I would like to thank my supervisor, Prof Bo An for providing guidance and support for this project. I would also wish to acknowledge the funding support for this project from Nanyang Technological University under the Undergraduate Research Experience on CAmpus (URECA) programme.



ICCC'19

June 17 — 21

Charlotte, North Carolina, USA

**10th International Conference
on Computational Creativity**