Welcome to the proceedings of the 12th International Conference on Computational Creativity (ICCC 2021), held from September 14 to 18, 2021. The original plan was for the conference to be held in México City, on the campus of the National University (UNAM, Universidad Nacional Autónoma de México) and the campus of the Universidad Autónoma Metropolitana (UAM)—Cuajimalpa. While last year's conference was the first in which there was a repeat of the host country (Portugal), this year would have been the first time to repeat the host city, albeit the host institution would have been different. However, the current Covid pandemic and its negative impact on international travel forced us to organize the conference virtually instead of having a physical meeting in México City.

We received 57 submissions for long (regular, normal) papers this year, ultimately accepting 32 of them (21 for oral presentation and 11 for poster presentation) based on the recommendations of the Program Committee members who reviewed them. Likewise, we received 71 submissions for short papers, ultimately accepting 30 (19 for oral presentation and 11 for poster presentation). Of the accepted papers, both long and short, 14 were earmarked by their authors to be presented as demos during the conference. The accepted papers originated in academic and research institutions from 23 countries (Australia, Austria, Belgium, Canada, Denmark, Finland, France, Germany, Greece, Ireland, Israel, Italy, Japan, México, the Netherlands, Portugal, Saudi Arabia, Slovenia, Spain, Taiwan, Turkey, the United States of America, and the United Kingdom).

In addition to the oral, poster, and demo presentations given at the conference based on the accepted papers, ICCC 2021 also included the following associated events:

- Four tutorials (Avatars for All!, organized by Eyal Gruss; Live Coding Music with Machine Learning in the Browser, organized by Francisco Bernardo, Chris Kiefer, and Thor Magnusson; Therapeutic Computational Creativity, organized by Louis McCallum and Mick Grierson; and Using Machine Learning to Build Musical Instruments in the Browser with MIMIC, organized by Maya Ackerman and Galen Buckwalter),
- One workshop (Second Workshop on the Future of Co-Creative Systems),
- An art exhibition (curated by Lilla LoCurto and Bill Outcault),
- An AI keynote lecture (Machine Learning Design for Creative Pursuits by Douglas Eck),
- Two "other perspectives" keynote lectures (Animal Innovation by Allison Kaufman and James C. Kaufman; and Comedy by Numbers: A Comedy Writer’s Thoughts on Algorithmic Approaches to Humour by Charlie Skelton),
- A panel on Opportunities and Challenges for Computational Creativity as a Commercial Application (moderated by F. Amílcar Cardoso, with Maya Ackerman, Allison Parrish, and Davar Ardalan as panelists),
- A Doctoral Consortium (organized by Diarmuid O’Donoghue),
- A tribute to Robert Keller (organized by Anna Jordanous and Maya Ackerman), and
- Various virtual social gatherings and meetings.

We would like to acknowledge the assistance of the Program Committee members (see below) who helped review the submissions, the student volunteers who helped out with multiple tasks...
before and during the conference, as well as the following individuals without whose help we would not have been able to organize ICCC 2021:

- Constanza Ybarra Trapote, Proceedings Preparation Support
- Carlos G. Isaac, Technical and Artistic Support
- Diarmuid O'Donoghue, Doctoral Consortium Chair
- Caleb Rascón, Workshops and Tutorials Chair
- Jesús Pérez Romero, Media Chair

In any given year, organizing the ICCC is both a great responsibility and a great privilege. While the responsibility is perhaps more obvious than the privilege, the latter more than compensates for the occasionally frenzied moments of the former. As organizers and chairs, we get to see the conference take shape for another year, and observe the deep currents that shape our field rise to the surface as waves of one kind or another. We are privileged with an early glimpse of how the field is moving and maturing, as it reflects—or bucks—the trends that shape other branches of AI and Computer Science. Papers are rated by reviewers not by how well they reflect these trends, but on their own individual merits, so it is fascinating to see how representative the final selection is of the whole. This year we can see the field balance at the edge of two shifting plates of AI research: the “old school” AI of symbols, rules and explicit knowledge representations, and the “new school” AI of statistics, big data and deep learning. The shift to the latter is evident in all branches of AI, but ICCC still retains a fondness and a respect for the former. This is a field and a community where the two can sit cheek-by-jowl and where researchers, and systems, can potentially learn from each other.

One especially bittersweet aspect of an event like this is the need to reject papers that do not make the cut, not because they are lacking in merit but because we cannot accommodate them all in the tight schedule of a three-day conference. The short paper format allows for some of these papers to find their way into the conference in a condensed form, one that benefits from the reviewers’ comments of the first round, so we are heartened to see the conference enriched by the work of authors who availed of this possibility. This is another privilege of being an organizer of ICCC. We get to see the review process in full flow, to take the pulse of the field by reading not just the papers but all of their reviews too, for those that make the cut and those that do not, and for those that find a second life as short papers and system demonstrations. We feel confident that when you see the lineup of papers that the reviewers have collectively chosen for us this year—for it is the reviewers, not the program chairs, that ultimately make all the key decisions—you too will agree that our field is in rude good health.

Andrés Gómez de Silva Garza, Program Co-Chair—Instituto Tecnológico Autónomo de México (ITAM)
Tony Veale, Program Co-Chair—University College, Dublin
Wendy Aguilar, Local Chair—Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS), Universidad Nacional Autónoma de México (UNAM)
Rafael Pérez y Pérez, General Chair—Universidad Autónoma Metropolitana (UAM) Cuajimalpa

México City/Dublin, September 2021
Program Committee

Program Committee Members:

The listed affiliations, sourced from EasyChair, are assumed to be correct at the time these proceedings were prepared.

Alan Tapsco—Universidad Complutense de Madrid
Allison Parrish—New York University
Amy K. Hoover—New Jersey Institute of Technology
Ana Rodrigues—University of Coimbra
Ana-Maria Olteteanu—Human-Centered Computing, Freie Universität Berlin
Andrés Gómez de Silva Garza—Instituto Tecnológico Autónomo de México (ITAM)
Anna Jordanous—University of Kent
Anna Kantosalo—Aalto University
Antonios Liapis—Institute of Digital Games, University of Malta
Ashok Goel—Georgia Institute of Technology
Bob Sturm—KTH Stockholm
Carlos León—Universidad Complutense de Madrid
Christian Guckelsberger—Finnish Centre for Artificial Intelligence, Aalto University
Colin Johnson—University of Nottingham
Dan Ventura—Brigham Young University
Daniel Brown—University of Waterloo
David Meredith—Aalborg University
David C. Brown—CS Dept., WPI
Diarmuid O'Donoghue—Maynooth University
Duri Long—Georgia Institute of Technology
Enric Plaza—IIIA-CSIC
F. Amílcar Cardoso—University of Coimbra
Fabrizio Góes—University of Leicester
Fabrizio Balducci—University of Bari
Gabriella A. B. Barros—Modl.ai
Georgios N. Yannakakis—Institute of Digital Games, University of Malta
Geraint Wiggins—Vrije Universiteit Brussel / Queen Mary University of London
Hannu Toivonen—University of Helsinki
Heather Ligler—Georgia Institute of Technology
Hugo Gonçalo Oliveira—University of Coimbra
Ivan Guerrero—UNAM-IIMAS
Jer Hayes—Accenture
João Correia—University of Coimbra
João Miguel Cunha—University of Coimbra
Jon McCormack—Monash University
Julian Togelius—New York University
Kazjon Grace—UNC Charlotte
Kirvanç Tatar—Simon Fraser University
Lonce Wyse—National University of Singapore
Margareta Ackerman—Florida State University
Maria Navarro—BISITE
Maria Teresa Llano Rodriguez—Monash University
Martin Znidarsic—Jožef Stefan Institute
Matthew Guzdial—University of Alberta
Maximos Kaliakatsos-Papakostas—School of Music Studies, Aristotle University of Thessaloniki
Michael Cook—Queen Mary University of London
Mikhail Jacob—Microsoft Research
Milena Fisher—The Creativity Post
Nada Lavrač—Jozef Stefan Institute
Nuno Lourenço—University of Coimbra
Oliver Bown—Interactive Media Lab, UNSW
Oliver Kutz—KRDB Research Centre for Knowledge and Data, Free University of Bozen-Bolzano
Pablo Gervás—Universidad Complutense de Madrid
Pablo de la Torre—University of Cádiz
Paul Bodily—Idaho State University
Pedro Martins—University of Coimbra
Penousal Machado—University of Coimbra
Phil Lopes—HEI-Lab, Universidade Lusófona
Philipp Wicke—University College Dublin
Philippe Pasquier—Simon Fraser University
Rafael Pérez y Pérez—Universidad Autónoma Metropolitana at Cuajimalpa, México
Raquel Hervás—Universidad Complutense de Madrid
Roisin Loughran—DkIT
Sarah Harmon—Bowdoin College
Senja Pollak—University of Ljubljana
Simo Linkola—University of Helsinki
Tapio Takala—Aalto University
Tatsuo Unemi—Soka University
Tony Veale—University College Dublin
Wendy Aguilar—IIMAS, Universidad Nacional Autónoma de México
ICCC 2021 Art Exhibition

Curators: Lilla LoCurto and Bill Outcault

Exhibits:

David Young
  ten images, *Dandelion series*, 2020

David Rokeby

Alexander Reben
  one video, *Five dollars can save the planet*, 2018

Ellen Pearlman
  one video, *Noor-A Brain Opera*, 2016

Jeroen Van derMost
  one video and three images, *The Shard Transformation*, 2021

Current Team (Collaborators: Provides Ng, Eli Joteva, Ya Nzi, and Artem Konesvkikh)
  one video, *Current*, 2020

Derek Curry and Jennifer Gradecki
  one video, *Infodemic*, 2021

Ken Goldberg and the AlphaGarden Collective
  one video, *AlphaGarden*, 2020

Mary Flanagan
  one video, *Grace [AI]*, 2019

Liliana Farber
  eleven images, *Terram in Aspectu*, 2019

Jake Elwes
  interactive website, *The Zizi Show*, 2020

Catie Cuan
  one video, *OUTPUT*, 2018

Sofian Audry and Maurizio Maritnucci (akaTeZ)
Contents

1 Art & Aesthetics ............................................. 1
  1.1 Computational Filling of Curatorial Gaps in a Fine Arts Exhibition
      Arthur Flexer ........................................... 2
  1.2 Visual Conceptual Blending with Large-Scale Language and Vision Models
      Songwei Ge and Devi Parikh .......................... 6
  1.3 What Does it Take to Cross the Aesthetic Gap? The Development of Image Aesthetic
      Quality Assessment in Computer Vision
      Samuel Goree ........................................... 11
  1.4 Framing through Music: A Pilot Study
      Stephen James Krol, Teresa Llano and Cagatay Goncu ........... 16

2 Linguistic Creativity .................................... 21
  2.1 Witscript: A System for Generating Improvised Jokes in a Conversation
      Joe Toplyn ............................................. 22
  2.2 Structures in Tropes Networks: Toward a Formal Story Grammar
      Jean-Peic Chou and Marc Christie ..................... 32
  2.3 Conceptualizing Human-Computer Intersubjectivity to Develop Computational Hu-
      mor
      Henna Paakki ........................................... 42
  2.4 Inspiration through Observation: Demonstrating the Influence of Automatically Gen-
      erated Text on Creative Writing
      Melissa Roemmele ....................................... 52
  2.5 Exploring a Masked Language Model for Creative Text Transformation
      Hugo Goncalo Oliveira .................................. 62
  2.6 Nightmare Machine: A Large-Scale Study to Induce Fear Using Artificial Intelligence
      Pinar Yanardag, Nick Obradovich, Manuel Cebrian and Iyad Rahwan ........... 72
  2.7 Survival of the Wittiest: Evolving Satire with Language Models
      Thomas Winters and Pieter Delobelle .................... 82
  2.8 Evaluating Natural Language Descriptions Generated in a Workspace-Based Archi-
      tecture
      George Wright and Matthew Purver .................... 87
  2.9 Assessing MultiPlot Stories: From Formative Analysis to Computational Metrics
      Pablo Gervás, Eugenio Concepcion and Gonzalo Mendez .......... 92
  2.10 Collaborative Storytelling with Human Actors and AI Narrators
      Boyd Branch, Piotr Mirowski and Kory Mathewson ............. 97

3 Musical Creativity ....................................... 102
  3.1 MuSyFI—Music Synthesis from Images
      André Santos, H. Sofia Pinto, Rui P. Jorge and Nuno Correia .......... 103
  3.2 Supporting Computational Music Remixing with a Co-Creative Learning Companion
      Erin J.K. Truesdell, Jason Brent Smith, Sarah Mathew, Gloria Ashiya Katuka,
      Amanda Griffith, Tom McKlin, Brian Magerko, Jason Freeman and Kristy Elizabeth
      Boyer .................................................... 113
3.3 LyricJam: A System for Generating Lyrics for Live Instrumental Music
Olga Vechtomova, Gaurav Sahu and Dhruv Kumar ........................................ 122

4 Social Dimensions 131
4.1 Walking the Line in Computational Creativity: Paradox and Pitfall in the Mitigation
of Automated Offense
Tony Veale .................................................................................................................. 132
4.2 Scientific Question Generation: Pattern-Based and Graph-Based RoboCHAIR Meth-
ods
Senja Pollak, Vid Podpečan, Janez Kranjc, Borut Lesjak and Nada Lavrač ............... 140
4.3 Inspiring Social Creativity in Children with CUBUS
Patrícia Alves-Oliveira, Raquel Oliveira, Patrícia Arriaga, Ana Paiva and Carlos
Martinho ...................................................................................................................... 149
4.4 Are Machine Learning Corpora “Fair Dealing” under Canadian Law?
Daniel Brown, Lauren Byl and Maura Grossman ..................................................... 158
4.5 The Stone of Madness Meets AI
Rafael Pérez y Pérez, Lilla Lo Curto and Bill Outcault ........................................... 163
4.6 The 7th AISB Symposium on Computational Creativity (CC@AISB2020/21)
Anna Jordanous and Juan Alvarado ........................................................................... 168

5 Theory & Philosophy 172
5.1 Incorporating Algorithmic Information Theory into Fundamental Concepts of Com-
putational Creativity
Tiasa Mondol and Daniel Brown .................................................................................. 173
5.2 Creativity and Consciousness: Framing, Fiction and Fraud
Geraint Wiggins .......................................................................................................... 182
5.3 Embodiment and Computational Creativity
Christian Guckelsberger, Anna Kantosalo, Santiago Negrete-Yankelevich and Tapio
Takala .......................................................................................................................... 192

6 Theory & Practice 202
6.1 Should Machines Evaluate Us? Opportunities and Challenges
Kuakou Bossou and Maya Ackerman ........................................................................... 203
6.2 Producing Creative Chess through Chess Engine Selfplay
Wolf De Wulf .............................................................................................................. 208
6.3 Meta-Evaluating Quantitative Internal Evaluation: A Practical Approach for Devel-
opers
Filippo Carnovalini, Nicholas Harley, Steven T. Homer, Antonio Roda and Geraint
A. Wiggins ................................................................................................................... 213
6.4 Software Design Patterns of Computational Creativity: A Systematic Mapping Study
Porter Glines, Isaac Griffith and Paul Bodily ............................................................. 218
6.5 Deciphering The Cookie Monster: A Case Study in Impossible Combinations
Maria Hedblom, Guendalina Righetti and Oliver Kutz .............................................. 222

7 Tools & Techniques 226
7.1 Active Divergence with Generative Deep Learning–A Survey and Taxonomy
Terence Broad, Sebastian Berns, Simon Colton and Mick Grierson ......................... 227
7.2 Generative Search Engines: First Experiments

Simon Colton, Amy Smith, Sebastian Berns and Ryan Murdock ................. 237

8 Visual Creativity .......... 247

8.1 Surprising Image Compositions

Othman Sbai, Camille Couprie and Mathieu Aubry ................................... 248

8.2 A Deep Learning Pipeline for the Synthesis of Graphic Novels

Thomas Melistas, Giannis Siglidis, Fivos Kalogiannis and Ilan Manouach ........ 256

8.3 A Case-Based Approach to Creating Movie Poster Compositions

Jorge Alejandro Ramírez Gallardo, Brandon Francisco Hernández Troncoso, Rodrigo Castillo Gómez and Andrés Gómez de Silva Garza ......................... 266

8.4 MIST: “You Play, I’ll Draw”

Juliana Shibadeh and Maya Ackerman ................................................ 276

8.5 GANlapse Generative Photography

Simon Colton and Blanca Perez Ferrer ................................................... 281

8.6 Adversarial Learning of Expectation and Surprise: Experiments with Geometric Shapes

Oseremen Uduehi and Razvan Bunescu .................................................. 286

9 Poster Papers .......... 291

9.1 When a Computer Cracks a Joke: Automated Generation of Humorous Headlines

Khalid Alnajjar and Mika Hämäläinen ...................................................... 292

9.2 Ukiyo-e Analysis and Creativity with Attribute and Geometry Annotation

Yingtao Tian, Tarin Clanuwat, Chikahiko Suzuki and Asanobu Kitamoto .......... 300

9.3 Reflective Creators

Max Kreminski and Michael Mateas ...................................................... 309

9.4 Metaphor, Blending and Irony in Action: Creative Performance as Interpretation and Emotionally-Grounded Choice

Tony Veale and Philipp Wicke ............................................................. 319

9.5 Conceptual Expansion Neural Architecture Search (CENAS)

Mohan Singamsetti, Anmol Mahajan and Matthew Guzdial ............................ 327

9.6 Punch Card Knitting Pattern Design in Collaboration with GAN

Virginia Melnyk ................................................................. 336

9.7 Shape Inference and Grammar Induction for Example-Based Procedural Generation

Gillis Hermans, Thomas Winters and Luc De Raedt .................................. 342

9.8 Syllable Neural Language Models for English Poem Generation

Danielle Lewis, Andrea Zugarini and Eduardo Alonso ............................. 350

9.9 Automating Generative Deep Learning for Artistic Purposes: Challenges and Opportunities

Sebastian Berns, Terence Broad, Christian Guckelsberger and Simon Colton ... 357

9.10 Entrepreneurship: A New Frontier in a Computational Science of Creativity

Ashok Goel, Mukundan Kuthalam, Sung Hong and Keith McGregor ............ 367

9.11 Ideation via Critic-Based Exploration of Generator Latent Space

Punctet Jain, Najma Mathema, Jonathan Skoggs and Dan Ventura .................. 377

9.12 Learning to Rank Generated Portmanteaus

Lara Pollet, Thomas Winters and Pieter Delobelle .................................... 386
<table>
<thead>
<tr>
<th>Page</th>
<th>Title</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>391</td>
<td>SOVIA: Sonification of Visual Interactive Art</td>
<td>Lauryn Gayhardt and Maya Ackerman</td>
</tr>
<tr>
<td>395</td>
<td>Creating new Program Proofs by Combining Abductive and Deductive Reasoning</td>
<td>Kuruvilla George Aiyankovil, Diarmuid O’Donoghue and Rosemary Monahan</td>
</tr>
<tr>
<td>400</td>
<td>In the Name of Creativity: En Route to Inspiring Machines</td>
<td>Chun-Yien Chang and Ying-Ping Chen</td>
</tr>
<tr>
<td>405</td>
<td>CREA.blender: A GAN Based Casual Creator for Creativity Assessment</td>
<td>Miroslav Gajdacz, Janet Rafner, Steven Langsford, Arthur Hjorth, Carsten Bergenholtz, Michael Biskjaer, Lior Noy, Sebastian Risi and Jacob Sherson</td>
</tr>
<tr>
<td>410</td>
<td>Dynamic Creation of Points of View</td>
<td>Alessandro Valitutti</td>
</tr>
<tr>
<td>415</td>
<td>Composition, Performance and Evaluation: A Dance Education Framework for AI Systems</td>
<td>Sarah Ibrahimi</td>
</tr>
<tr>
<td>419</td>
<td>Towards a Visual Language Using Neural Networks</td>
<td>Luís Gonçalo, João Miguel Cunha and Penousal Machado</td>
</tr>
<tr>
<td>424</td>
<td>A Large-Scale Computational Model of Conceptual Blending Using Multiple Objective Optimisation</td>
<td>João Gonçalves, Pedro Martins and F. Amâncar Cardoso</td>
</tr>
<tr>
<td>429</td>
<td>Wölfflin’s Affective Generative Analysis for Visual Art</td>
<td>Divyansh Jha, Hanna Chang and Mohamed Elhoseiny</td>
</tr>
<tr>
<td>434</td>
<td>The Humble Creative Machine</td>
<td>Christopher Cussion, Maya Ackerman and Anna Jordanous</td>
</tr>
</tbody>
</table>

10 Appendix: Proceedings of the Second Workshop on the Future of Co-Creative Systems

<table>
<thead>
<tr>
<th>Page</th>
<th>Title</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>439</td>
<td>Preface: Second Workshop on the Future of Co-Creative Systems</td>
<td></td>
</tr>
<tr>
<td>440</td>
<td>Evaluating the Effect of Co-Creative Systems on Design Ideation</td>
<td>Jingoog Kim and Mary Lou Maher</td>
</tr>
<tr>
<td>444</td>
<td>COFI: A Framework for Modeling Interaction in Human-AI Co-Creative Systems</td>
<td>Jeba Rezwana and Mary Lou Maher</td>
</tr>
<tr>
<td>449</td>
<td>Agency in Co-Creativity: Towards a Structured Analysis of a Concept</td>
<td>Janin Koch, Prashanth Thattai Ravikumar and Filipe Calegario</td>
</tr>
<tr>
<td>453</td>
<td>AI-Aided Co-Creation for Wellbeing</td>
<td>Haizi Yu, James Evans, Donna Gallo, Adam Kruse, William Patterson and Lav Varshney</td>
</tr>
</tbody>
</table>
1. Art & Aesthetics
Computational filling of curatorial gaps in a fine arts exhibition

Arthur Flexer
Institute of Computational Perception
Johannes Kepler University Linz, Austria
arthur.flexer@jku.at

Abstract

We present an interactive computational approach to fill curatorial gaps between artworks in a fine arts exhibition. We describe the algorithmic details of our semantic approach based on word embedding of keywords and how we include additional curatorial constraints. We present an installation at a museum exhibition and discuss lessons learned during our arts based research.

Introduction

This work is part of the arts-based research project “Dust and Data: the Art of Curating in the Age of Artificial Intelligence”\(^1\), where we explore how machine learning can help both curators and audiences discover and navigate large museum collections. In a previous workshop contribution (Flexer 2020) we presented an approach to compute smooth semantic pathways between works of art. In this paper we report about an installation at our project exhibition “Dust and Data - Artificial Intelligence im Museum” at the Austrian Museum of Folk Life and Folk Art in Vienna, Austria, adapting and extending this approach. Our installation asks the question about curatorial gaps between artworks shown in an exhibition. What works of art exist in the holdings of the museum that fit the curatorial narrative but did not succeed in becoming part of the exhibition?

Our approach is inspired by the project “X Degrees of Separation”\(^2\) by “Google Arts and Culture”, which explores the “hidden paths through culture” by analyzing visual features of artworks to find pathways between any two artifacts through a chain of artworks. While we find these pathways aesthetically pleasing, we are, from a curatorial perspective, more interested in finding pathways of the semantic meaning of works of art. We chose this semantic driven approach because vital information about a piece of art cannot be found in the artwork itself. Think e.g. of subjecting the “Mona Lisa” to an automatic visual analysis. Computational results will tell you that it is a picture of a young woman, in front of a landscape, and (if your algorithm is really good) is sort of smiling. This information of course totally misses the significance of the painting for (Western) art history, its immense relevance and the many connotations it has. All of this rather is a societal construct and result of centuries of discourse and reception history. Another problem with analysing the visual content of paintings is that state-of-the-art image analysis is usually trained on photographs and generalization to paintings is not trivial, especially for more abstract artforms (Kim et al. 2019).

As a consequence, we chose to use word embedding (Mikolov et al. 2013) to embed keywords of a museum collection and obtain pathways through the resulting semantic space as well as to add curatorial semantic constraints. Word embedding encodes semantic similarities between words by modelling the context to their neighboring words in a large training text corpus.

Data

We obtained 4365 artworks and their keywords from Belvedere’s (Vienna, Austria) online collection\(^3\), representing art from the middle ages to the present time. We exclude all sculptures and three-dimensional art, keeping 3421 artworks which are mainly paintings and drawings.

To demonstrate our curatorial approach of filling gaps, we chose part of a room in Belvedere’s permanent exhibition. It is a room about “Viennese portraiture in the Biedermeier period”, assembling the “greatest portrait painters” from this period. The three paintings marked (a), (d) and (g) in figure 1 are part of the original exhibition. Artworks in between ((b), (c), (e) and (f)) are proposed by our algorithm described in the next section.

Methods

Semantic embedding: The online collection is indexed with 3585 different keywords which are often very specialized with a fourth of them being assigned only once. Since this keyword index is therefore rather sparse and of only limited help for organization of the collection, we use natural language processing to compute similarities between keywords thereby obtaining a semantic embedding space. Specifically we use the tool spacy\(^4\) to remove stop words and extract only nouns, this e.g. changes the original keyword ‘scepter, ruling staff (as symbol of highest force)’ into the four keywords ‘scepter’, ‘ruling staff’, ‘symbol’ and ‘force’. Since

\(^1\)http://www.dustanddata.at/
\(^2\)https://artsexperiments.withgoogle.com/xdegrees/
\(^3\)https://sammlung.belvedere.at/
\(^4\)https://spacy.io/
some of the artworks are not indexed by any keywords, we gain additional keywords by applying the same procedure to the titles of artworks also. This results in 6216 different keywords for which word embeddings actually exist. Please note that we translate all keywords from German to English for this paper. We use the German fasttext⁵ word embedding, which has been trained on about 3 million words from the Wikipedia- and 19 million words from the Common Crawl-corpus (Mikolov et al. 2018). This gives us a vector representation of size 100 for every keyword, with the cosine between vectors indicating semantic similarity. A cosine of 1 signifies perfect semantic similarity and 0 no similarity at all. To obtain a similarity \( \cos(a, b) \) between any two artworks \( A_a \) and \( A_b \) with \( k_a \) and \( k_b \) keywords, we simply average all possible crosswise cosine distances between keyword lists.

**Curatorial semantic constraint:** Looking at the keywords of the three paintings from the original exhibition (marked (a), (d) and (g) in figure 1), one can see that most of them are purely descriptive, e.g. ‘headgear’, ‘necklace’, ‘bonnet’, ‘eye contact’, probably not doing the semantic content of the artworks full justice. We also believe that one underlying semantic topic of the Biedermeier room is ‘gender’, with all but one painting depicting females. We therefore add an additional algorithmic constraint by requiring all suggested artworks to respect both the requirement of being part of a pathway and having a ‘gender’ related keyword. Since ‘gender’ is not a keyword in the Belvedere taxonomy, we use word embedding to obtain Belvedere keywords with high similarity to the topic of ‘gender’. This translation step yields the following top ranking keywords with cosine similarity between 0.60 and 0.45: Islam, religion, headscarf, education, blinder, equal opportunities, religions, hacking, asylum, robe, female labor, feminalness, fan, skirt, medicine, ornament, force, avowal, psychiatry, delusion, blindness, doctrine. There are a number of keywords in the same high similarity range which we excluded from this list for being too general: non-, context, science, instrument, conversation, points of view, attribute, natural sciences. There are 132 artworks with at least one of these keywords in the database of 3421 mostly paintings and drawings.

**Choosing artworks:** To compute artworks to fill the gap between a start artwork \( A_s \) and an end artwork \( A_e \), we do the following for a database of \( n \) artworks \( A_i \):

1. for all \( i = 1, \ldots, n \) artworks compute cosine similarities to start artwork \( \cos(i, s) \) and end artwork \( \cos(i, e) \)
2. find \( m_n \) artworks with largest similarity \( \cos(i, s) \) to \( A_s \); find \( m_n \) artworks with largest similarity \( \cos(i, e) \) to \( A_e \); keep only \( m = |m_n \cup m_n| \) artworks for further processing
3. for all \( i = 1, \ldots, m \) artworks compute a similarity ratio:
   \[
   R(i) = \frac{\cos(i, s)}{\cos(i, e)}
   \]
4. order all \( i = 1, \ldots, m \) artworks according to their similarity ratio \( R(i) \)

Artworks which are closer to start artwork \( A_s \) than to end artwork \( A_e \) will have a similarity ratio \( R(i) > 1 \), while those closer to \( A_e \) than to \( A_s \) will have \( R(i) < 1 \). Artworks which have equal similarity to both \( A_s \) and \( A_e \) will have a similarity ratio \( R(i) \) around 1. Since this is also true for artworks with small but equal similarity, they have to be excluded from the pathway as outliers, keeping only the \( m_n \) artworks closest to either \( A_s \) or \( A_e \) in step 2. For our revision of the Biedermeier room we chose \( m_n = m_n = 30 \). Values greater than 30 yielded artworks with too little similarity.

**Results**

In figure 1, the three paintings marked (a), (d) and (g) are part of the original exhibition, with the other four paintings being one specific solution obtained with the algorithm described above. These artworks at positions (b), (c), (e) and (f) fill the curatorial gaps by forming a transition between (a) and (d), and (d) and (g) respectively. At the same time each of the obtained artworks obeys the curatorial constraint of having at least one of the keywords from the ‘gender’ list. This constraint results in only 133 artworks from 3421 being eligible for the solution. In step 2 of our algorithm we keep the \( m_n = m_n = 30 \) closest artworks in consideration for our solution between artworks (a) and (d), and (d) and (g) respectively. Removing duplicates in these lists via \( m = m_n \cup m_n \) leaves us with 18 artworks to fill the gap between (a) and (g), and 19 artworks between (d) and (g). From these lists of 18 or 19 artworks we can now randomly select two artworks each to fill the curatorial gaps. Since in step 4 of our algorithm these artworks are ordered according to their similarity ratios, every selection needs to respect this ordering to obey the requirement of a smooth transition. For \( f = 18 \) artworks, there are \((f(f-1))/2 = 153\) possibilities to choose two artworks respecting the ordering. For \( f = 19 \) artworks there are 171 possibilities.

Returning to figure 1, we would now like to discuss this exemplary solution out of the many possible ones. Painting (b) is suggested because its keyword ‘femaleness’ (in bold face in figure 1) is a gender keyword and its keyword ‘necklace’ makes it similar to the keywords of painting (a) (‘earrings’, ‘pearl necklace’) and to painting (d) (‘brooch’, ‘bracelet’). Keyword ‘portrait of a girl’ is also related to ‘portrait’ of painting (a). All these similarities together explain why painting (b) fits into the transition between (a) and (d) while at the same time being related to the concept of ‘gender’.

Similar arguments can be given for the other filling artworks. Painting (c) has the gender keyword ‘fan’ which also appears for (a). It also has the keyword ‘inner room’ (and ‘church interior’) which also appears for (d). Painting (e) has the gender keyword ‘force’, but also ‘empire’ and ‘emperor’ which all relate to keyword ‘princess’ of (d). Keyword ‘eye contact’ appears also for (d) and (g). Painting (f) has the gender keyword ‘head scarf’ which connects to the keyword ‘feather hat’ of (d). Keyword ‘eye contact’ appears for (d), (f) and (g). The keyword ‘spouse’ also connects to ‘woman’ and ‘lady’ of painting (g).

Leaving the discussion about the one specific solution in figure 1, it is interesting to see which gender keywords appear for all 18 plus 19 possible transition artworks. Keyword ‘head scarf’ appears 18 times, ‘fan’ 8 times, ‘force’ 5 times, ‘skirt’ 3 times, ‘femaleness’, ‘religion’, ‘ornament’ and ‘avowal’

⁵https://fasttext.cc/
Once, all others never. It is noteworthy that the majority of keywords (‘headscarf’, ‘fan’, ‘skirt’) describe apparel or accessories. These keywords are of course closer to the many keywords of the original Biedermeier paintings (a), (d) and (g) also describing apparel or accessories. It would be interesting to keep the curatorial constraint of requiring a ‘gender’ keyword for every filling artwork, but excluding these keywords from the computation of similarity between artworks. This could result in gender related artworks more detached from keywords describing apparel.

Somehow problematic are keywords describing pictorial organization (Bove, Heusinger, and Kailus 2016) like ‘multiple layer room’, since the word embedding is not able to grasp such subtle semantics. Names of historical persons as keywords (e.g. ‘Chorinsky’ or ‘Esterházy’) are often not adequately embedded in the semantic space and very general terms like ‘men’ or ‘figure’ can also lead the algorithm astray.

Returning to the full list of gender related keywords which we obtained via word embedding, it is also striking that many keywords point to a stereotypical discourse of gender, quickly derailing towards topics of ‘religion’ and ‘Islam’ and a compulsion to wear ‘headscarfs’, or a discussion of ‘femaleness’ and ‘force’, probably pointing to women still being subjected to violence in today’s society. This is also why we like to term the use of word embedding in this context world embedding: it confronts the very rigid taxonomy of the Belvedere keywords (based on Iconclass⁶, a multilingual classification system for cultural content) with everyday language as represented in the textual training data of the word embedding. It thereby re-contextualizes or even re-socializes taxonomic art histories via natural language processing since it uncovers biases and prejudice in our use of language and (re-)introduces them to the world of fine arts.

The actual installation of the Biedermeier room revision was part of our project exhibition “Dust and Data - Artificial Intelligence im Museum” at the Austrian Museum of Folk Life and Folk Art in Vienna, Austria. The installation is a half-scale copy of part of Belvedere’s Biedermeier room and can be seen in figure 2. The three original paintings (positions (a), (d) and (g) in figure 1) are shown as reproductions printed on linen to set them apart from the artworks at the curatorial gap positions. Artworks selected by our algorithm are projected at their respective positions including keyword information. Please note that to keep keyword information readable, only single artworks are projected between (a) and (d), and (d) and (g) respectively. The artworks are shown in a repeated random order, realizing a sort of flickering representing the many artworks that fit the curatorial gaps but have not been shown by the original curation.

Conclusion

We have presented an approach to compute pathways between works of art that also follow an overarching curatorial constraint, enabling audiences to discover transitions based on semantic content instead of visual information. There are three lessons we have learned while building the art installation based on this technology described in our paper:

(i) Generally speaking, semantic approaches should be more helpful for building a curatorial narrative (Wolff, Mulholland, and Collins 2012) than a purely aesthetic procedure. After all, museum curation relying on visual information only is hard to conceive. Our specific procedure allows to

---

⁶http://www.iconclass.org/
answer the question about curatorial gaps between artworks shown in an already existing exhibition.

(ii) Using a machine learning tool like word embedding, curating becomes a joint endeavor of man and machine, where curatorial decisions have to be formulated as input and constraints to the algorithm. As such we tried “to design programs that can enhance human creativity without necessarily being creative themselves”, which is one of the goals of computational creativity. A fact which is hardly ever discussed is that even a simple curatorial Google search already is an interaction of man and machine, with algorithms to a certain extent (oblique to the curator) shaping their curatorial enterprise by showing specific selections of information only. All these man/machine approaches are able to uncover algorithmic biases in the methods used, as e.g. stereotypical representations of societal discourse in word embedding.

(iii) Looking towards future extensions of our work it can be said that of course we could analyse longer (art historic) texts about artworks with the same methodology thereby gaining much richer semantic context then by relying on simple keywords only. Another possible extension is to embed semantic and visual information simultaneously which could yield curatorial solutions that respect semantic and visual constraints at the same time (Frome et al. 2013; Kim et al. 2019).

Acknowledgments
This work is funded by the Austrian Science Fund (FWF): AR532. We gratefully acknowledge cooperation and support by Belvedere Research Center.

References
Visual Conceptual Blending with Large-scale Language and Vision Models

Songwei Ge\textsuperscript{1} and Devi Parikh\textsuperscript{2}
\textsuperscript{1}University of Maryland, College Park
\textsuperscript{2}Facebook AI Research & Georgia Tech

Abstract

We ask the question: to what extent can recent large-scale language and image generation models blend visual concepts? Given an arbitrary object, we identify a relevant object and generate a single-sentence description of the blend of the two using a language model. We then generate a visual depiction of the blend using a text-based image generation model. Quantitative and qualitative evaluations demonstrate the superiority of language models over classical methods for conceptual blending, and of recent large-scale image generation models over prior models for the visual depiction.

Introduction

Throughout the development of human civilization, our unique capacity to blend unfamiliar concepts has led to innovation of advanced tools, invention of new art styles, and breakthroughs in science. Machines demonstrating this ability is considered to be one of the hallmarks of creativity and intelligence. Such systems could help understand human creativity. Moreover, they can assist humans in exploring the inexhaustible space of combinations of different concepts. This has been an area of research for decades (Fauconnier and Turner 1998), which has led to both theoretical work (Cunha, Martins, and Machado 2020) as well as prototypes of support tools to assist users (Karimi et al. 2018; Chilton, Petridis, and Agrawala 2019). In the meantime, deep learning has achieved exceptional success in many areas where humans excelled, from beating the best professional player(Silver et al. 2016) in Go to making creative advertising designs(Brown et al. 2020).

In this paper, we examine deep neural networks trained on large-scale data in a general scenario of visual conceptual blending: given a single object as input (e.g., moon), can a relevant object [that is conceptually grounded (Cunha, Martins, and Machado 2020)] be identified (e.g., an orange), can a relevant property that a blend can hinge on be identified (e.g., sliced), and finally, can an image be generated to depict the blend (e.g., “the moon sliced like an orange”)? We use prompt-engineering with language models for the reasoning phase (identifying a relevant object and property), and text-based image generation models for the visualization phase. See Figure 1 for example outputs.

![Example Outputs](a) “A tree made of blue and red blood vessels”. (b) “The moon that is sliced like an orange”.

Figure 1: Visual conceptual blends generated by our framework using large-scale language and vision models.

We compare our approach quantitatively and qualitatively to representative existing approaches. To evaluate the ability to associate concepts, we compare our approach to traditional knowledge bases on a simile dataset. To evaluate the visual generation, we compare our approach to an existing GAN approach via human studies. We show that large-scale models significantly outperform these baseline models. In general, we find that an appropriate composition of recent large-scale models results in encouraging creative abilities like visual conceptual blending.

Related Work

Visual Conceptual Blending Fauconnier and Turner first proposed the idea of conceptual blending and pointed out its indispensability in human development (Fauconnier and Turner 1998; 2008). Cognitive and neural scientists have been fascinated by the human ability to blend concepts and view such an ability as a milestone for AI development (Eppe et al. 2018). More practically, the idea of visual conceptual blending has been applied in many commercial areas from advertising, journalism, to public service announcements (Chilton, Petridis, and Agrawala 2019). In this section, we discuss the recent progress in developing systems that automatically blend visual concepts and the studies that measure the success of conceptual blending.

[Computational approaches to conceptual blending such as Divago (Pereira and Cardoso 2006) and COINVENT (Schorlemmer et al. 2014; Eppe et al. 2018) follow]
the seminal idea based on Mental Spaces Theory (Fauconnier 1994)]. Many systems developed by these studies act as support tools for augmenting human creativity. (Chilton, Petridis, and Agrawala 2019) [present] a workflow where users identify the associated concepts, retrieve appropriate images, and label the analogous parts of the objects while the system automatically blends the images by combining these common parts. (Vismantic (Xiao, Linkola, and others 2015) on the other hand retrieve and pre-process the images for given words, ask a human to pick ideal photos, and automatically combine the images in fixed ways.] (Karimi et al. 2018) [explore] visual conceptual blends in the context of sketching by leveraging the idea of concept shifts. (Cunha et al. 2017) [propose] a description-based method that can blend sketches using detailed annotations. See (Cunha, Martins, and Machado 2020) for a road map of visual conceptual blending. (McCaig, DiPaola, and Gabora 2016; Berov and Kuhnberger 2016) [apply] style transfer models and the deep dream algorithm to render an image in a particular artistic style. (Sbai, Couprie, and Aubry 2021) [study] placing objects in uncommon contexts using a search-and-compose method. Measuring the creativity of visual blends is known to be difficult. Fauconnier and Turner proposed several optimality principles to guide the conceptual blending (Fauconnier and Turner 1998). (Martins et al. 2015) [analyze] what makes a good blend using 15 hybrid animal images and a questionnaire.

**Analogical Reasoning with Language Models** Language models were first proposed to model the sequential nature of language (Mikolov and Zweig 2012). With the increasing sizes of training data and model capacities, large-scale language models such as BERT (Devlin et al. 2018) fine-tuned on the downstream tasks have dominated standard leaderboards. Interestingly, several recent studies use language models as knowledge bases to solve different problems without training on the task of interest (Petroni et al. 2019; Jiang et al. 2020). These methods rely on task-specific prompts – converting the task of interest to that of language modeling. Letting the language model predict masked parts from the prompt then becomes equivalent to the model solving the task of interest (Petroni et al. 2019; Jiang et al. 2020). We propose to apply a similar idea to concept blending – we design appropriate prompts to identify relevant concepts and properties along which to blend the concepts. Analogical reasoning has also been approached with large-scale knowledge bases (Liu, Wu, and Yang 2017). However, knowledge bases are known to be incomplete and rigid. We argue that this makes them less suitable for associating concepts in flexible ways (Cunha, Martins, and Machado 2020).

**Deep Generative Models for Images** Most state-of-the-art image generation methods are built on either Generative Adversarial Networks (GANs) (Goodfellow et al. 2014) or Variational AutoEncoders (VAEs) (Kingma and Welling 2014). In this paper, we are primarily interested in generating conceptually blended objects. (Bau et al. 2020) [propose] to modify the images through manipulating the intermediate layers in GANs which admits the possibility to blend concepts. In this work we use a textual description of the blend to guide the generation. Text-based image generation models (Reed et al. 2016; Zhu et al. 2019; Tao et al. 2020) are relevant. DALL·E (Ramesh et al. 2021) is one such recent model that uses a pretrained discrete VAE to compress images into low-dimensional vectors and then models the joint distribution of the vectors with text embeddings autoregressively.

**Table 1:** Top 5 concepts relevant to *moon*, and associated properties using simile-inducing prompts to a BERT model.

<table>
<thead>
<tr>
<th>concept</th>
<th>property</th>
</tr>
</thead>
<tbody>
<tr>
<td>ghost</td>
<td>dead</td>
</tr>
<tr>
<td>dream</td>
<td>over</td>
</tr>
<tr>
<td>rainbow</td>
<td>broken</td>
</tr>
<tr>
<td>beacon</td>
<td>colorful</td>
</tr>
<tr>
<td>jewel</td>
<td>lit</td>
</tr>
</tbody>
</table>

**Approach**

Next, we describe how we use large language and image generation models to produce conceptually blended images given an input object. We decompose the visual conceptual blending process into two phases: reasoning and generation.

**Reasoning Phase.** 1 The reasoning phase produces a textual description of the blend. We formulate the problem as follows: given an input object, the model identifies a relevant object and generates a description of the blend of the two. Note that our setting is more general than one where both concepts to be blended are given as input (Cunha, Martins, and Machado 2020). [We explore two prompt engineering approaches, simile-inducing and property-guided prompts, which connect the input objects to other objects that are either generally relevant, or in terms of a specific property.] We use *moon* as the example input to explain the details of our prompt engineering approach.

To identify a relevant object, we use a simile-inducing input: “the moon is like a [MASK]” and ask the language model to predict the masked word. The language model produces *ghost*, i.e. “the moon is like a ghost”. Next, we utilize the prompt “the ghost has the property of [MASK]”, where the language model predicts the word *dead*. We plug the predictions into a template and produce the description of the blend “a moon that is dead like a ghost”. Other concepts and their properties identified using a pretrained BERT (Devlin et al. 2018) model are shown in Table 1. Sometimes the retrieved objects are semantically similar rather than visually similar to the *moon* such as *ghost* and *dream*. We see some interesting blends such as “a moon that is lit like a beacon” and “a moon that is broken like a rainbow”.

Shape is often recognized as the bridge to connect different visual concepts (Steinbrück 2013; Chilton, Petridis, and Agrawala 2019). This motivates a shape-guided prompt to identify relevant objects. Specifically, we first use language

---

1An example colab notebook of the reasoning process can be found at [https://colab.research.google.com/drive/1A6jO AchESHn_kX7G7tVdS9hZfa8Zxp5R?usp=sharing](https://colab.research.google.com/drive/1A6jO AchESHn_kX7G7tVdS9hZfa8Zxp5R?usp=sharing).
models to predict the shape of the moon with the prompt “The shape of the moon is [MASK1]”. The language model outputs *spherical*. This gives us “The shape of the moon is spherical”. Then we plug the word *spherical* into the prompt “The shape of the [MASK2] is spherical”, and the language model predicts the relevant object shell, i.e. “The shape of the shell is spherical”. This leads to a blend description “a moon that is smooth like a shell” with the property *smooth* of the shell. More identified concepts and their properties are shown in Table 2. We find that the candidate concepts we obtain are visually similar to the moon in terms of shape. Some interesting descriptions include “a moon that is laid like an egg” and “a moon that is edible like a fruit”. In practice, shape can be replaced by other properties that connect visual concepts. For example, speed connects *bullet* and *runner* and reflection connects *mirror* and *lake*.

*Generation Phase* In this phase we generate an image based on the description output by the reasoning phase. To demonstrate the ability of large models in realizing the blends, we explore BigSleep\(^2\) and DeepDaze\(^3\) which utilize the CLIP model (Radford et al. 2021) to guide the BigGAN (Brock, Donahue, and Simonyan 2019) and SIREN (Sitzmann et al. 2020) models for text-based image generation.

Specifically, suppose we are given a trained CLIP model \(f_B(x_i, x_s)\) which takes an image \(x_i\) and a sentence \(x_s\) as input and outputs the similarity, and a trained BigGAN model \(g_θ(z)\) which takes a random Gaussian vector \(z\) as input and outputs an image. We first sample a vector \(z_0\) from a standard Gaussian distribution. \(z\) is iteratively updated to maximize the similarity of the generation \(g_θ(z)\) and the text \(x_s\) as computed by the CLIP model \(f_B(x_i, x_s)\). DeepDaze adopts a similar process with BigGAN replaced by SIREN.

Overall, we now have a full pipeline to go from an input concept (e.g., *moon*) to a description of its blend with a related concept (e.g., “a moon that is sliced like an orange”) to an image that depicts this blend (e.g., Figure 1).

### Table 2: Top 5 concepts relevant moon, and associated properties using shape-guided prompts to a BERT model.

<table>
<thead>
<tr>
<th>concept</th>
<th>property</th>
</tr>
</thead>
<tbody>
<tr>
<td>shell</td>
<td>white smooth</td>
</tr>
<tr>
<td>head</td>
<td>rounded black</td>
</tr>
<tr>
<td>fruit</td>
<td>edible white</td>
</tr>
<tr>
<td>egg</td>
<td>white yellow</td>
</tr>
<tr>
<td>eye</td>
<td>open closed</td>
</tr>
</tbody>
</table>

|                               | smooth thin brown small |
|                               | black white brown small |
|                               | yellow red purple      |
|                               | laid blue red          |
|                               | small red black        |

Figure 2: Visual blends generated using different methods using blend descriptions shown at the bottom as input.

*Evaluation*

*Reasoning Phase* To evaluate how well language models blend concepts, we evaluate on the simile dataset (Chakrabarty, Muresan, and Peng 2020). It contains pairs of literal input and its simile version in the form of \(<\text{Source}, \text{Target}>\), e.g. \(<\text{The city was beautiful}, \text{The city was like a painting}>\). It evaluates the model’s ability to identify “painting” based on “the beautiful city”. However, we found that the language is inconsistent across the dataset. For instance, many pairs lack a subject or use a pronoun as subject, e.g. \(<\text{Felt worthless}, \text{Felt like a low budget film}>\). We instead focus our evaluation on the model’s ability to accomplish the core reasoning step — predicting the property “worthless” based on the object “a low budget film”. Using heuristics for pre-processing, we extracted 66, 442 property-objects pairs for evaluation.

We compare language models to knowledge bases. For the language model we use the prompt “a low budget film is [MASK]” as the input and ask the model to generate candidate predictions for the masked word. We consider 4 trained language models: ELMO (Peters et al. 2018), BERT\(_{\text{Base}}\) and BERT\(_{\text{Large}}\) (Devlin et al. 2018), and GPT (Radford et al. 2018). For knowledge base, we use ConceptNet (Speer, Chin, and Havasi 2017) which contains relations including “IsA”, “HasA”, “HasAProperty”, etc., which form candidate predictions for properties relevant to the object.

Note that sometimes the object in our dataset is described as a phrase including qualifiers (e.g., “a low budget film”) while ConceptNet only contains the root objects. We use dependency parsing to find the root of the phrase and use it to query ConceptNet. In our example, “film” instead of “a low budget film” is used. After this processing, 96.34% of objects from our evaluation set can be found in ConceptNet.

For each method, we produced 1000 candidates, and report the precision, i.e. percentage of time that the property (e.g., “worthless”) is in the top 10, 100, 1000 candidates. Note that the ConceptNet API does not offer a straightforward way to request an exact number of relations for an object. Different objects have different number of properties associated with them. When requesting 1000 relations for objects in our evaluation set, 688.90 were returned on average. As shown in Table 3, the precision using ConceptNet is significantly lower than using language models.

Additionally, we notice that using larger language models can further improve the precision. In general, these results demonstrate that language models are better at associating concepts than knowledge bases. We hypothesize this is due...
Figure 3: Human preference for different methods w.r.t. different questions. Values outside the band between the dashed lines are statistically significant at 95% confidence.

Table 3: Precision of language models and knowledge base on the simile dataset.

<table>
<thead>
<tr>
<th></th>
<th>P@10</th>
<th>P@100</th>
<th>P@1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConceptNet</td>
<td>1.12</td>
<td>2.70</td>
<td>5.90</td>
</tr>
<tr>
<td>Elmo</td>
<td>0.13</td>
<td>7.69</td>
<td>37.33</td>
</tr>
<tr>
<td>BERTbase</td>
<td>1.59</td>
<td>15.72</td>
<td>53.08</td>
</tr>
<tr>
<td>BERTlarge</td>
<td>1.42</td>
<td>15.89</td>
<td>46.56</td>
</tr>
<tr>
<td>GPT</td>
<td>2.59</td>
<td>24.84</td>
<td>66.38</td>
</tr>
</tbody>
</table>

Generation Phase We collect 20 text descriptions of blends (see Figure 2 for examples) – half generated with our reasoning approaches and rest by us. We use these descriptions as input to the large-scale BigSleep and DeepDaze models described earlier, as well as a recent DF-GAN (Tao et al. 2020) model. We run human evaluation on Amazon Mechanical Turk (AMT). We show subjects a pair of images generated by different methods to depict the visual blend of two objects and ask six questions: 1. In which image do you recognize OBJECT1 more? 2. In which image do you recognize OBJECT2 more? 3. Which image blends the two objects better? 4. Which image conveys the DESCRIPTION better? 5. Which image looks more interesting to you? 6. Which image looks more aesthetically pleasing to you? These are designed using the optimality principles for concept blending [(Fauconnier and Turner 1998; Cunha, Martins, and Machado 2020)]. Specifically, 1 and 2 relate to the unpacking principle, 3 and 4 to the integration principle, and 5 and 6 to general quality. Each question (6) for every pairwise comparison of models (3) and every textual description (20) is answered by 9 unique subjects.

See results in Figure 3. The CLIP-based models (BigSleep and DeepDaze) significantly outperform DF-GAN, demonstrating the superiority of large models in generating visual blends. BigSleep is preferred over DeepDaze. We conjecture that this is because BigGAN learns a better prior on the image distribution than SIREN.

Conclusion

In this paper, we apply large-scale language and image generation models to a classic computational creativity problem – visual conceptual blending. Our experiments show that these models allow us to use simple yet effective ways to generate visual blends that are significantly better than previous methods. Future work includes engineering novel prompts to connect concepts and developing more complex blending strategies given the identified concepts. For example, the classic blend of boat and house (houseboat) – “a man lives in a house that is built on the water like a boat” – considers structural relationships of the objects and includes two different properties from the two objects – a place of accommodation (from house) and being on water (from boat).

References


Brock, A.; Donahue, J.; and Simonyan, K. 2019. Large scale gan training for high fidelity natural image synthesis.


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. ICCC.

Cunha, J.; Martins, P.; and Machado, P. 2020. Let’s figure this out: A roadmap for visual conceptual blending. In ICCC.


Jiang, Z.; Xu, F. F.; Araki, J.; and Neubig, G. 2020. How can we know what language models know? TACL.

Karimi, P.; Maher, M. L.; Grace, K.; and Davis, N. 2018. A computational model for visual conceptual blends. IBM JRD.

Kingma, D. P., and Welling, M. 2014. Auto-encoding variational bayes. ICLR.
Liu, H.; Wu, Y.; and Yang, Y. 2017. Analogical inference for multi-relational embeddings. In ICML. PMLR.


Speer, R.; Chin, J.; and Havasi, C. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In AAAI.


What Does it Take to Cross the Aesthetic Gap? The Development of Image Aesthetic Quality Assessment in Computer Vision

Sam Goree
Department of Informatics
Indiana University
Bloomington, IN
sgoree@iu.edu

Abstract
Computer vision research into image aesthetic quality assessment seeks to use machine learning to measure the aesthetic quality of images, which bears resemblance to a variety of topics within computational creativity, but has not been discussed in our community. To foster a conversation around this research literature, we trace the development of computer vision algorithms for aesthetic judgment over the past fifteen years and critically consider whether these algorithms actually cross the “aesthetic gap” proposed by the first researchers in this space. We then build towards a more fundamental question regarding machine learning and subjectivity.

Introduction
Over the past fifteen years, computer vision researchers have investigated algorithms for image aesthetic quality assessment (IAQA). This research area seeks to apply machine learning to measure the aesthetic quality of images, usually by classifying them as “beautiful” or “not beautiful.” While it is tempting to dismiss such a task as hopeless, given the subjectivity of the problem, this research has applications in automatic photo editing and curation and relates to several topics within computational creativity, including Machado and Cardoso’s computing aesthetics (1998), Greenfield’s computational aesthetics (2005), and Fisher and Shin’s computational criticism (2019).

An interesting concept which arises from this literature is the notion of the aesthetic gap. Roughly analogous to the semantic gap in information retrieval, which separates the low-level features of images like pixels and lines from the high-level features humans observe in images like objects and symbols (Hare et al. 2006), Datta, Li, and Wang (2008) define the aesthetic gap as separating “the information that one can extract from low-level visual data” and “the interpretation of emotions that the visual data may arouse in a particular user.” Since aesthetics is central to the value of many creative artifacts, and such value is often seen as an essential component of creativity (Boden 2004), thinking about the aesthetic gap and whether our algorithms really cross it is of central importance to computational creativity.

In this “debate spark” paper, however, we turn the concept of the aesthetic gap back on IAQA and question whether recent progress in this field actually constitutes a cross over that gap. To explore this topic, we present a historical narrative tracing the development of computer vision methods to measure aesthetic quality and their relation to prior philosophical and psychological study of aesthetics and its measurement. By introducing this problem area to the ICCC community, we hope to generate interest in developing computational approaches to aesthetics which engage more substantially with the fundamental subjectivity of the problem.

Quantifying Aesthetics Before Computing
Before digging into the IAQA literature in computer vision, however, we will briefly define the term “aesthetics,” give an example of how it is used in philosophy and discuss two influential historical attempts to measure the aesthetic qualities of stimuli.

In philosophy, aesthetics is the study of beauty, taste, experience and judgment. While many questions in philosophical aesthetics have a long history, even dating back to Plato, the study of aesthetics, as a “science of perception” is first named in the 18th century work of Alexander Gottlieb Baumgarten, whose work inspired responses from enlightenment philosophers (Guyer 2007). These philosophers understood that aesthetics is highly subjective, and developed methods for overcoming this subjectivity.

Immanuel Kant, in particular, had an influential approach. He claimed that judgments differed from person to person because they were bound up with interest, meaning that we make judgments based on feelings of pleasure, not just reason. However, if we remove our interests and make judgments which are completely disinterested, we can make judgments of taste, which should be universal among rational people. Kant is quick to clarify, though, that just because disinterested judgment is universal does not mean that such judgments can be made objectively, or based on the object alone: “aesthetic universality...does not unite the predicate of beauty with the concept of the object...and yet extends it to the whole sphere of judging persons,” (Kant 1790).

Taste varies from person to person, across time and place and is even highly subject to influence, even in a laboratory setting (Bignardi, Ishizu, and Zeki 2020). Despite these challenges, aesthetics is one of the oldest topics of study in psychology, dating back to the 19th century work of the experimental psychologist Gustav Fechner. Fechner showed 347 subjects a series of rectangles and ellipses and asked...
them to choose the most appealing, and the rectangle with proportions drawn from the golden ratio was chosen the most frequently (Green 1995).

Fechner’s work on aesthetics has been criticized by later psychologists and philosophers. For example, the 20th century Gestalt psychologist Rudolf Arnheim identifies a connection between Fechner’s interest in measuring perception of beauty with his larger spiritual, cosmological and philosophical beliefs, and argues that Fechner’s view of beauty as something which can be distilled down to one variable makes his findings related to art scientifically unreliable. “Just as Fechner’s study does not tell us why people prefer the ratio of the golden section to others, so most of the innumerable preference studies carried out since his time tell us deplorably little about what people see when they look at an aesthetic object, what they mean by saying that they like or dislike it, and why they prefer the objects they prefer,” (Arnheim 1985).

Inquiry specifically into aesthetic measures, like the ones put forward by contemporary computer vision researchers, starts with the work of the 20th century American mathematician George Birkhoff. Birkhoff’s 1933 book, Aesthetic Measure puts forward a theory of aesthetic experience which divides it into three phases: first we recognize the complexity of a work, next we feel the sense that it is valuable, then finally we recognize the underlying order to which it adheres. Birkhoff claims these three properties: order (O), complexity (C) and value (M), can be related via an equation $M = O^C$.

Birkhoff’s approach, like Fechner’s, has been extremely influential, inspiring a century of computational approaches to aesthetics (e.g. Moon and Spencer’s model of color harmony (Moon and Spencer 1944)), but it is poorly regarded by many philosophers. For example, Susanne Langer claims that the easily described nature of musical harmony has led to a great deal of hope that other aspects of art might be quantified and understood mathematically as well. However, “there is no use discussing the sheer nonsense or the academic oddities to which this hope has given rise, such as...the serious and elaborate effort of G.D. Birkhoff to compute the exact degree of beauty in any art work (plastic, poetic and musical) by taking the ‘aesthetic measure’ of its components and integrating these to obtain a quantitative value judgment,” (Langer 1953). Langer goes on to argue that while musical sound is easy to describe, such description does not access the artistic qualities of music like motion, which exist in virtual space and time, rather than in the physical sound.

Langer’s criticism of Birkhoff invokes a similar criterion to Datta, Li, and Wang: the difference between the explicitly measurable qualities of an object and the virtual and experiential qualities which inform its aesthetics are quite similar to the idea of a semantic or aesthetic gap. While rather simplistic mathematical models like those of Birkhoff likely lack the capacity to model something comparable to a human’s aesthetic response, it is unclear whether more sophisticated computer vision models learned from data share that limitation.

**Early Machine Learning Approaches**

Contemporary study of aesthetics in computer vision begins with the simultaneous work of Datta et al. and Ke, Tung, and Jing in 2006. Despite both working at the same time, and in the same US state (Pennsylvania), these two groups of authors arrived the problem area from different conceptual directions and take different approaches within the context of image classification.

Datta et al. are determined to automatically learn from...
data which factors influence aesthetic value. They claim that, “in spite of the ambiguous definition of aesthetics...there exist certain visual properties which make photographs, in general more aesthetically beautiful.” (Datta et al. 2006) Their concept of aesthetic value originates from their data: over 3000 images collected from the website photo.net, which allows users to upload their photos, and allows other users to rate them on “aesthetics” and “originality.”† They cite two other sources on their understanding of aesthetics: the Oxford Advanced Learner’s Dictionary and a book, Rudolf Arnheim’s 1965 Art and Visual Perception. A Psychology of the Creative Eye (1965). Aesthetic quality assessment is framed in terms of image classification: they train decision trees and support vector machines to classify images into high and low aesthetics categories based on a variety of features extracted from images (e.g. measures of colorfulness, the photographic rule-of-thirds, image dimensions).

The decision to cite Arnheim pulls this approach towards psychological aesthetics, a field which exists in dialogue with both the work of earlier psychologists like Fechner, as well as the history of aesthetic philosophy. In a later survey paper (Joshi et al. 2011), the same authors cement that link. They discuss the approaches of analytic philosophers like Nelson Goodman and Richard Wollheim, as well as recent work in neuroaesthetics by Semir Zeki, who claims that aesthetic experience can be identified and explained by activity in specific brain regions.

To contrast, Ke, Tang, and Jing (2006) approach IAQA from the perspective of photo curation. Rather than psychological aesthetics, they ground their work in image quality assessment, an area of computer vision research concerned with measuring image noise and degradation (Kamble and Bhurchandani 2015). Rather than making claims about philosophy, Ke, Tang, and Jing argue that a well-designed set of features may be used to reason about the subjective aspects of image quality, like the difference between professional and amateur photos. Their method makes use of images and ratings from the photo challenge website DPChallenge.com, which they divide into “professional” and “amateur” categories based on ratings. They cite two popular photography books to justify their choices of features, which include edge and color histograms, as well as Fourier transform-based blur metrics, which they use to train a Naive Bayes classifier.

Over the next six years, a variety of other publications emerged proposing different combinations of image features for solving the aesthetic quality assessment problem. While other scholars used similar approaches at first (Datta, Li, and Wang 2008; Jiang, Loui, and Ceresaletti 2010), later authors shifted towards low-level features like GIST or SIFT descriptors due to an influential paper by Marchesotti et al. which made the case that hand-crafted features are ineffective because they are non-exhaustive, computationally expensive and rely on heuristic assumptions which may not generalize well (Marchesotti et al. 2011).

The relationship between Datta, Ke and both earlier and later aesthetic thought is at the heart of our claims about the aesthetic gap. The work of Datta et al. is framed as an approach to computational aesthetics, but like Ke, Tang, and Jing, they only measure how consistent a photograph is with common photography rules of thumb. Later work further conflates these two concepts of “aesthetic quality” by shifting to lower-level image features to better fit the dataset labels. However, inspection of the “high quality” and “low quality” images in these datasets makes it clear that the distinction between them is more of a stylistic difference than anything else. Figure 1 shows comparisons between high and low quality photos from two IAQA papers. The qualities shared by all of the photos labeled as “high quality” is evident: these are overwhelmingly photos of landscapes and flowers which prioritize color and emotionality over realism. We would argue, however, that this style is not the only way that photographs can be beautiful. Photography can be aesthetically pleasing in as many ways as other art forms, and many genres of art photography like candid photography or photojournalism do not prioritize the use of such dramatic visual effects. In other words, these papers and datasets seem to conflate explicit emotionality with the potential to arouse emotion.

The AVA Dataset and Deep Learning

In 2012, two major events shifted the conversation around IAQA. First, in June, Murray, Marchesotti, and Perronnin (2012) released the Analysis of Visual Aesthetics (AVA) dataset, which contains over 250,000 photos from DPChallenge.com, an order of magnitude larger than any existing dataset, along with metadata, including rating distributions and category labels, where possible. Second, in October, Krizhevsky, Sutskever, and Hinton (2012) dramatically beat the benchmark on the ImageNet LSVRC using a deep convolutional neural network (CNN). While deep learning had profound effects on computer vision as a whole, these two contemporaneous changes produced a paradigm shift in the study of IAQA.

Lu et al. (2014) published the first paper applying deep learning to aesthetic image classification in 2014. They reiterate the argument from Marchesotti in favor of generic image features, and claim that deep features are even more generic, since they work with pixels directly. Lu et al. identify that the fixed input size of AlexNet makes it difficult to apply to images of many different dimensions in AVA, since cropping or warping might disrupt aesthetic quality, so they use a two-column model to learn from warped and cropped versions of the image simultaneously. Neither this work, nor the generation of papers which followed their lead in applying CNNs to the AVA dataset (Kao, Wang, and Huang 2015; Zhou et al. 2016; Lv and Tian 2016), make much reference to the problem statement and its context at all, aside from acknowledging its highly subjective nature.

While CNNs do not carry all of the assumptions of things like measures of colorfulness or edge histograms, they are not blank slates either. The connectivity structure of convolutional and max-pooling layers encode the as-
sumption that the salient features of an image are locally situated, translation-invariant and the presence of an activation is more significant than the absence, which are good assumptions for classifying between different types of objects or handwritten digits (Krizhevsky, Sutskever, and Hinton 2012), but are not necessarily good for aesthetic judgment, which at least in the eyes of psychologists like Arnheim (1965), is more holistic.

In the past five years, several trends have emerged in IAQA. First, Kong et al. (2016) suggest including user data to personalize image assessments, which Ren et al. (2017) formalize into an active learning task. Second, different objectives beyond classification have emerged, including pairwise comparison (Lv and Tian 2016) and distribution learning (Cui et al. 2017). Finally, the binary classification accuracy benchmark on the AVA dataset has steadily increased, reaching over 91% (see Table 1).

Additionally, a new claim for significance, related to cur- ration and editing of photographs for social media, has emerged. Several recent authors make reference to the widespread popularity of social networking services (Wang et al. 2019), the exponential growth of online visual data (Sheng et al. 2018; Lee and Kim 2019) and the growing need for automatic photo editing tools (Wang et al. 2019). This claim for significance brings IAQA into the realm of AI-based creativity support tools, further increasing its relevance to the computational creativity community.

Our narrative in this section emphasizes the continuity between the current state of the art in IAQA and the long history of aesthetics in other disciplines. There is a direct continuity from classical to deep methods: Marchesotti et al. made their argument in favor of low-level image features before the advent of deep learning, and the first deep learning-based method of Lu et al. is framed as the natural extension of that argument. Even highly technical recent papers, which are quite distant from the philosophical motivations of authors like Dutta et al., are implicitly weighing into a long conversation on the nature of art and beauty, which may have wide reaching implications. But do any of them really cross the aesthetic gap and reason about “the interpretation of emotions that the visual data may arouse in a particular user?”

### Table 1: Accuracy benchmark results on the AVA dataset.

<table>
<thead>
<tr>
<th>Paper</th>
<th>Year</th>
<th>Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Murray, Marchesotti, and Perronnin 2012)</td>
<td>2012</td>
<td>67%</td>
</tr>
<tr>
<td>(Lu et al. 2014)</td>
<td>2014</td>
<td>71%</td>
</tr>
<tr>
<td>(Mai, Jin, and Liu 2016)</td>
<td>2016</td>
<td>77.1%</td>
</tr>
<tr>
<td>(Kong et al. 2016)</td>
<td>2016</td>
<td>77.3</td>
</tr>
<tr>
<td>(Zhou et al. 2016)</td>
<td>2016</td>
<td>78.1%</td>
</tr>
<tr>
<td>(Wang et al. 2016)</td>
<td>2016</td>
<td>76%</td>
</tr>
<tr>
<td>(Kao, He, and Huang 2017)</td>
<td>2017</td>
<td>78%</td>
</tr>
<tr>
<td>(Ma, Liu, and Wen Chen 2017)</td>
<td>2017</td>
<td>82.5%</td>
</tr>
<tr>
<td>(Ko, Lee, and Kim 2018)</td>
<td>2018</td>
<td>82.2%</td>
</tr>
<tr>
<td>(Sheng et al. 2018)</td>
<td>2018</td>
<td>83.3%</td>
</tr>
<tr>
<td>(Lee and Kim 2019)</td>
<td>2019</td>
<td>91.5%</td>
</tr>
</tbody>
</table>

Discussion

So far, we have traced the evolution of IAQA in computer vision from its prehistory in the work of Fechner and Birkhoff, its origins in psychological aesthetics and photographic rules of thumb and its shift from hand-engineered features to deep learning. We saw how its two goals, rooted in computational aesthetics and image quality assessment, merged over time, and how performance on the AVA dataset, which arguably only captures a specific, popular photographic style, has been treated as a stand-in for an algorithm’s ability to measure aesthetic quality more generally. With that continuity in mind, we find it difficult to point to a specific paper or accuracy level where these approaches cross the aesthetic gap introduced by Datta, Li, and Wang. However, such a claim raises other questions about the nature of this gap.

For example, it’s possible that the success of recent deep learning models on the AVA dataset demonstrates that there is no such gap: the neuroscientific arguments indicate that our aesthetic responses exist in a lower level of the visual system than we might believe (Chatterjee and Vartanian 2016), and it’s possible that we actually make judgments based on simple visual statistics and only use higher cognitive processes to explain those judgments. Such a finding would vindicate scholars like Birkhoff, who believed that a measure of aesthetics could be computed from measures of order and complexity, without regard for the emotions of the observer. On the other hand, if we assume that an aesthetic gap does exist, and making aesthetic judgments requires algorithms which understand meaning and emotional attachment, that would cast further doubt on whether IAQA models are actually measuring aesthetics, and whether accuracy on the AVA is a suitable measure of performance.

If deep learning models cannot overcome the aesthetic gap, how should we, as artificial intelligence researchers, proceed? It’s not unreasonable to imagine a computationally creative agent which both interprets symbols and models emotional attachments enough to have something resembling an understanding of taste. But since taste is subjective, it is still unclear how to measure performance. Can a model have its own preferences, or should it merely predict the preferences of a human?

This last point reaches towards an important question regarding artificial intelligence and subjectivity. When we say that a task is subjective, who should be the subject? Is our goal to develop algorithms which have their own aesthetic experiences (for some definition of “own”), or merely predict the preferences of humans? If it is the former, is an algorithm a Kantian disinterested agent? If it is the latter, which humans’ preferences should count? Versions of this question exist throughout computational creativity. For example, should an algorithmic musician create music that appeals to its own computational sense of taste, its creator’s taste or an array of other humans’ tastes? IAQA chooses to derive ground-truth labels from an average of many humans’ aesthetic quality ratings, but such data risks conflating aesthetic quality with popularity. The theory and research methods for issues relating to aesthetics and subjectivity in machine learning demand more scholarly attention.
References


Abstract

We present a system that automatically generates music from visual art based on the perceived emotion of the given input. We propose the generated music as a framing device that can enhance the aesthetic experience of people viewing Computational Creativity (CC) outputs. In this paper, we carry out a first study to test this by comparing the aesthetic experience of viewing paintings generated by CC systems accompanied by either textual framing, our proposed musical framing or both. We evaluate our system by means of qualitative user evaluations, which require participants to rank their aesthetic experience from best to worst. The results from the study demonstrated that the musical framing generated by our system provided a better aesthetic experience for users compared to the textual framing. Furthermore, the results suggest that with more work, a combination of textual and musical framing could be used to further improve the aesthetic experience for people viewing visual CC art.

Introduction

Framing is an important element of creative work. In some cases, the framing of a creative output can increase its value and perceived creativity (Charnley, Pease, and Colton 2012). Traditionally, framing has been defined as describing how a creative process works. This is done using text as it is an effective medium to convey the narrative associated with a creative artefact. However, Gross et al. (Gross et al. 2014) expanded this definition to define framing as including anything created with a creative artefact that aims to change how the work or creator is perceived. This allows for other mediums to be used as framing devices, such as music. In this paper we introduce a system that aims to automatically generate emotive music from visual art. It does this by predicting the perceived emotion of the painting using two CNNs in the Valence-Arousal scale, then using this to find a piece of music from the VGMIDI database (Ferreira and Whitehead 2019) that elicits a similar emotion. This piece of music is then taken as inspiration to generate music in a similar style using Magenta’s Music Transformer (Huang et al. 2018). We present this system as a framing tool that aims to improve the aesthetic experience of viewing CC artwork by automatically generating musical framing that conveys the emotion of the painting.

Background

The concept of framing was first introduced in CC by Colton, Charnley and Pease in (Colton, Charnley, and Pease 2011) as “a piece of natural language text that is comprehensible by people, which refers to a non-empty subset of generative acts”. Simply put, framing is a device that has been used in CC to provide a description of how a program works, to explain its inputs or outputs and to provide insights about intrinsic factors behind the creative process behind it. Since its introduction, most works in CC that have used framing have done so in the form of textual commentaries attached to the creative output describing intentions, motivations and sources of inspiration that have guided the creative process towards the accompanying output.

However, novel approaches to framing have also been proposed within the community. An illustrative example is the approach proposed by Cook and Colton (Cook and Colton 2018) for ANGELINA, a computationally creative game design system, in which framing is used to communicate the design process over a period of time, allowing people to be involved in the “development and growth during creation not just after the fact”, as put by the authors. In this case ANGELINA documents the design process in a lot of detail with information such as lists of tasks and projects, version history, notes on success or failure, etc. and uses this
information to frequently inform its users about it through twitter and through the system’s blog.

Additionally, initiatives on the use of different framing devices has also been put forward. For instance, Gross et al. (Gross et al. 2014) described how a computational process of poetry generation was framed by means of an abstract visualisation and then turned into paintings by an artist, while in (Cook et al. 2019), the authors shift the emphasis of framing from creative acts onto the audience that is engaging with the work, giving rise to a revised definition of framing as follows: “‘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”, and the authors specifically highlight how this revised definition does not only refer to natural language as the only mechanism for framing.

In our work we follow this revised definition and propose the use of music as a method for framing visual art. We argue that the textual framing used currently by CC systems serve an informative purpose but fail at providing an aesthetic experience that is intrinsic to the act of experiencing and engaging with art. As described in (Charnley, Pease, and Colton 2012), the textual framing usually attempts to answer very practical questions about generative acts, particularly “why did you do X; how did you do X; and what did you mean when you did X?”. In our work, we propose the use of music as an alternative (or complementary) framing device for visual art and argue that music can be used to convey meaning as well as to provide the audience a more engaging experience, which is ultimately one of the purposes of framing as described in (Cook et al. 2019).

System description

A diagram of our system can be seen in figure 1. The current version of our system aims to generate music in the following stages:

1. Attempt to classify the perceived emotion in the Valence-Arousal scale.
2. Find music annotated with similar Valence-Arousal values.
3. Use this music as a primer for the Music Transformer in Magenta’s library (Huang et al. 2018)

Emotion Classification

The first stage of the system involves classifying the perceived emotion of the input painting using the Valence-Arousal model. To achieve this, we trained two convolutional neural networks (CNN) using the WikiArt dataset (Mohammad and Kiritchenko 2018). WikiArt is a dataset containing 4000 pieces of art that have been emotionally annotated by various observers. To the best of our knowledge, no work has been done in training a CNN to predict valence and arousal values associated with a painting using WikiArt. As suggested by the authors, we used the AG4 WikiArt dataset in our system. This dataset attributes an emotion to a painting when more than 40% of the annotators have applied it. An important step in preparing the dataset for training was mapping the categorical emotions to their respective values in the Valence-Arousal model. This involved analysing psychology literature and determining a general consensus for each emotion. Notable works used in these mapping are (De Bruyne, De Clercq, and Hoste 2020), (Hussain et al. ), (Jin and Wang 2005), (Sellers 2013) and (Wang et al. 2021).

To train the CNNs we used Keras version 2.24-tf with a Tensorflow 2.1.0 backend. Both models utilise transfer learning off the InceptionV3 (Szegedy et al. 2015) network pretrained on ImageNet. The architecture of both models can be seen in figure 2. We decided to train two separate models: one to predict valence and the other to predict arousal. This was because the WikiArt dataset is biased towards positive valence and high arousal emotions. Splitting the model into two networks simplified the unbalanced problem allowing us to effectively use undersampling to ensure balanced training. The tanh activation function was used in the final layer of both networks to ensure that outputs were in the desired (-1, 1) range. The networks were trained using a GTX1080ti with CUDA 10.1. Both networks used the Adams optimiser with a learning rate of 0.01 for 50 epochs with batch size 64 using the MSE loss function. The training and testing losses can be seen in table 1.

<table>
<thead>
<tr>
<th></th>
<th>Valence Model</th>
<th>Arousal Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Loss</td>
<td>0.12</td>
<td>0.09</td>
</tr>
<tr>
<td>Testing Loss</td>
<td>0.13</td>
<td>0.11</td>
</tr>
</tbody>
</table>

Table 1: Model Loss Metrics

Annotated Music

The VGMIDI annotated database (Ferreira and Whitehead 2019) was used as inspiration for the music transformer. VGMIDI contains 95 tracks from various video games in midi format. These tracks have been emotionally annotated with the Valence-Arousal model by 30 participants, with the study using a total of 1425 annotators. Each annotated track is split into different measures allowing participants to annotate different parts of the songs. The authors clustered the annotations into three groups: positive, negative and noise. The cluster with the most variance was considered noise...
while the best cluster was the one with the most annotations. The best cluster was then used to generate the median valence and arousal score for the different measures in each track. We match our predicted valence and arousal scores to these median annotations to find a piece of music to prime the music transformer. The aim is for the music transformer to produce a piece of music with similar valence and arousal scores as the primer; therefore, the resultant music would convey the perceived emotion of the painting.

Music Transformer
Once an appropriate primer was found, Magenta’s music transformer was used to generate music in a similar style to the primer. The music transformer is an attention-based neural network that attempts to generate music with coherent long-term structure. Compared to previous models from Magenta, such as performanceRNN (Simon and Oore 2017), the music transformer generates music that is more likely to play in a similar style to that of the primer. Maintaining a consistent style to its’ input is necessary to ensure the generated music has a similar valence-arousal values as its primer; which in turn has a similar score to the artwork. We utilised a ported script (Bao 2020) from the Google Music Magenta, such as performanceRNN (Simon and Oore 2017), the music transformer generates music that is more likely to play in a similar style to that of the primer. Maintaining a consistent style to its’ input is necessary to ensure the generated music has a similar valence-arousal values as its primer; which in turn has a similar score to the artwork. We utilised a ported script (Bao 2020) from the Google Music Transformer notebook with melody_conditioned_model_16 weights, to automatically generate midi files using the Music Transformer.

Pilot study
Methodology
In order to compare the experience of viewing a painting with textual framing versus musical framing, we prepared a pilot study. The study involved participants completing a small survey that contained eight paintings. The paintings and associated framing text were all generated by The Painting Fool (Colton, Valstar, and Pantic 2008). In two cases, framing text was unavailable so the authors manually created the framing text in the same style as The Painting Fool.

Four paintings conveyed positive valence while the other four conveyed negative valence. Two example paintings used in the study, one for each valence category, are shown in Figure 3. For each painting, our system generates music that attempts to convey the general emotion of the painting. We then use the painting, framing music and framing text to create three different experiences: Experience 1 is the painting combined with the framing text. Experience 2 combines the painting with the generated music. Finally, experience 3 combines both the framing text and music with the painting. The participants are then asked to rank the options in order of best experience. A link to a playlist containing the examples can be found in the appendix.

To control for order bias, three different surveys were created. The order of the paintings in each survey was shuffled using the Fischer-Yates algorithm which creates unbiased permutations. Participants were then randomly assigned one of the three surveys. We also shuffled the order of framing options for each painting.

Participants were recruited by responding to a call for participation posted in social media groups as well as sent to various email lists.

Results
A total of 21 participants responded to the call for participation and completed the survey. Participants had general knowledge in music and visual art; however, none were experts in either field. It is worth noting that three of these participants had low vision.

Generally, participants voted that the music and painting provided the best experience. The second best experience was the combination of painting, music and text. Finally, the text and painting combination was generally voted as the least favourite experience. The results from the survey can be seen in figure 4. Low vision participants ranked the combination of painting, music and text as providing the best experience 66% of the time.

One participant stated that "the artist’s description was bland, judgemental and contained a lot of useless information". Another participant stated that the "artist’s captions need to explain more about the art or the emotion you receive from it.” These comments suggest that the automatic framing generated by The Painting Fool could be enhanced with additional information or that alternative ways of framing may be more effective at communicating certain aspects of the creative process.

Discussion
The results from the survey demonstrated how the music generated by our system provided a better aesthetic experience for the users compared to viewing the painting with the automatically generated framing text. While this illustrates the benefit of using our system as a framing device, it also highlights improvements that are necessary when using text as the framing device. The Painting Fool’s framing...
Figure 3: Example of paintings from the Painting Fool used in the study.

text is not emotional but rather informational, even though it tries to convey its intention based on its perceived mood. In comparison, the music does attempt to invoke an emotional response in the user. Furthermore, while the painting and music combination was generally voted as the best experience, it was closely followed by the painting, text and music experience. This suggests that with some improvements to the framing text, our system could be utilised with The Painting Fool to create a better experience for users. This would be preferred over just using music to frame the artwork as there would be framing information (such as the artist’s inspiration), that would not be effectively communicated through music.

An interesting application of our system would be utilizing it as a method to improve the accessibility of visual art, and visual CC outputs, for people with vision impairments. Although framing text provides useful information, it does not effectively convey an aesthetic experience associated with the art. This aesthetic experience is one the main reasons sighted people view visual art and it should also be available to the visually impaired. Low vision participants ranked the combination of painting, text and music as the best experience 66% of the time. We hypothesise that the combination of framing text and music provides a better overall experience for the visually impaired as it allows the users to access more information relating to the artwork. Users can access both framing information and an aesthetic experience.

Conclusions and Future Work

In this study we introduced a system that could generate emotive music from visual artwork. The system classifies the perceived emotion of the painting using two CNNs and then utilising both the VGMIDI database and Magenta’s music transformer, generates new music that conveys the emotion of the input painting. We proposed that this system could be used as a framing device for CC systems that create visual outputs rather than just using traditional text based framing. A study was conducted to test how the musical framing generated by our system affected the aesthetic experience of the participants viewing visual art generated by The Painting Fool. The results suggested that the musical framing generated by our system provided a better aesthetic experience than viewing the paintings with just the framing text provided. Furthermore, the results indicate that with some improvements to the framing text, our system could be combined with The Painting Fool to create a better experience associated with the artwork.

We see the benefits of alternative forms of framing going beyond its current use; for instance in order to make CC outputs more accessible. Future work will look into using this system to improve the accessibility of visual art for the visually impaired. This will involve adding more features to the system, such as including ambient sound effects of objects detected by the system within the painting. Compared to just using textual framing, music can create an aesthetic experience associated with the artwork for the visually impaired. Combining our system with framing text could significantly improve the accessibility of visual art for people with vision impairments.

Figure 4: Survey Results: Demonstrates how the participants ranked the different experiences. In this study, 1 was the highest rank and 3 was the lowest.
impairments. This would also add a dimension of explainability to the generated musical framing, a feature that has been identified as important for CC systems (Llano et al. 2020). By including information of the objects detected by the system in the music, audience members can better understand what the system ‘sees’ from the paintings and as a result, better understand how the system works.

In this study, we did not evaluate the music directly. However, future work could also involve investigating the novelty of the generated music compared to its inspiration. Furthermore, investigating using features other than valence and arousal to generate music, such as velocity or tone, could allow the system to better frame its input artwork.

Acknowledgments

Cagatay Goncu is supported by the Australian Research Council (ARC) grant DE180100057. We thank the Faculty of Information Technology at Monash University for their support and Professor Simon Colton for providing the paintings and framing texts from the Painting Fool that were used in this work.

References


Appendices

Link to playlist containing examples used in the study: https://youtube.com/playlist?list=PLJXhSHZOX4QyIZPUnjEjf0Ajs0j1xplO
2. Linguistic Creativity
Witscript: A System for Generating Improvised Jokes in a Conversation

Joe Toplyn
Twenty Lane Media, LLC
P. O. Box 51
Rye, NY 10580 USA
joetoplyn@twentylanemedia.com

Abstract
A chatbot is perceived as more humanlike and likeable if it includes some jokes in its output. But most existing joke generators were not designed to be integrated into chatbots. This paper presents Witscript, a novel joke generation system that can improvise original, contextually relevant jokes, such as humorous responses during a conversation. The system is based on joke writing algorithms created by an expert comedy writer. Witscript employs well-known tools of natural language processing to extract keywords from a topic sentence and, using wordplay, to link those keywords and related words to create a punch line. Then a pretrained neural network language model that has been fine-tuned on a dataset of TV show monologue jokes is used to complete the joke response by filling the gap between the topic sentence and the punch line. A method of internal scoring filters out jokes that don't meet a preset standard of quality. Human evaluators judged Witscript's responses to input sentences to be jokes more than 40% of the time. This is evidence that Witscript represents an important next step toward giving a chatbot a humanlike sense of humor.

Introduction
For decades, people have imagined conversing with artificial entities as if they were human, and even deriving companionship from them. Recent advances in conversational AI have brought us closer to those social robots.

But to get even closer, a non-task-oriented conversational system—a social chatbot—needs to exhibit a sense of humor (Dybal, Ptaszynski, Rzepka, and Araki 2009a). The presence of humor improves a chatbot’s performance and makes it seem more humanlike (Dybal, Ptaszynski, Rzepka, and Araki 2009b). Humor is especially important in non-task-oriented agents because their main purpose is to socialize with and entertain human users (Ptaszynski et al. 2010).

In addition to being able to recognize humor, an inviting conversational agent must also be able to generate it (Nijholt, Niculescu, Valitutti, and Banchs 2017). Even adding a simple pun-based joke generator to a chatbot can significantly improve its performance, quality, and likeability (Dybal, Ptaszynski, Higuchi, Rzepka, and Araki 2008).

However, existing systems for the computational generation of verbal humor have notable limitations. The virtual agents created by some large corporations only deliver canned jokes prewritten by humans (Nijholt et al. 2017). Other systems only generate self-contained jokes that aren't closely tied to a context (Amin and Burghardt 2020).

To be truly useful, a computational humor system needs to generate contextually integrated jokes about what's happening at the moment (Ritchie 2005). Such a system could improvise original and relevant joke responses to a user's utterances in a conversation, much as a witty human friend might.

One reason that existing joke generators fall short is that often they are not based on any overarching theory of linguistic humor (Amin and Burghardt 2020). Indeed, not many published theories of linguistic humor are detailed enough to serve as useful frameworks for devising computable algorithms (Ritchie 2009).

But a few theories do lead to algorithms, including the Surprise Theory of Laughter, proposed by a four-time Emmy-winning comedy writer in his book (Toplyn 2014). The Surprise Theory of Laughter says that we laugh when we’re surprised that an incongruity turns out to be harmless.

The Surprise Theory of Laughter shares with the Benign Violation Theory (McGraw and Warren 2010) the idea that a necessary condition for laughter is that the incongruity be harmless.

The Surprise Theory of Laughter is also similar to the Two-Stage Model for the Appreciation of Jokes and Cartoons (Suls 1972), which is an incongruity-resolution model of humor. In the first stage of the Two-Stage Model, the audience encounters an incongruity—the punch line. In the second stage, the audience finds a cognitive rule that explains how the punch line follows from the preceding part of the joke, thus making sense of the incongruity. The Surprise Theory of Laughter differs from the Two-Stage Model in that the former stresses the importance of surprise: the audience must make sense of the incongruity suddenly, and therefore be surprised, if they are to laugh.

The Surprise Theory of Laughter provides the theoretical foundation for algorithms that Toplyn (2014) created to write the sort of monologue jokes delivered on comedy TV shows. Those algorithms specify that a monologue joke...
has three parts: the topic, the angle, and the punch line. The three parts appear in the joke in the following order:
1. The **topic** is the statement that the joke is based on.
2. The **angle** is a word sequence that smoothly bridges the gap between the topic and the punch line.
3. The **punch line** is the word or phrase that results in a laugh. It is a surprising incongruity that turns out to be harmless because it actually follows from the topic. This incongruity must appear at the end of the joke (Attardo 1994; Suls 1972).

Prewritten, canned jokes are structurally the same as jokes improvised during a conversation (Attardo 1994). Therefore, a system to improvise a conversational joke can be based on the above three-part structure of a canned monologue joke in the following way: A user's utterance to the system can be treated as the joke's topic. The improvised response generated by the system can be treated as the angle and punch line. So replying to the user’s utterance with the system's response can complete a three-part joke that may amuse the user.

Toplyn (2014) also states that a three-part monologue joke can be written by taking certain steps in a particular order. I've distilled those steps into this **Basic Joke-Writing Algorithm:**

1. **Select a topic.** A good joke topic consists of one sentence that is likely to capture the attention of the audience for the joke.
2. **Select two topic handles.** The topic handles are the two words or phrases in the topic that are most responsible for capturing the audience’s attention. The topic handles must capture the audience's attention because the audience must remember them in order to make sense of a punch line that is based on them. To remember information, we have to pay attention to it (Aly and Turk-Browne 2017).
3. **Generate associations of the two topic handles.** An association is something that the audience for the joke is likely to think of when they think about a particular subject. An association could be, for example, a person, place, thing, action, adjective, event, concept, or quotation.
4. **Create a punch line.** The punch line links an association of one topic handle to an association of the other topic handle in a surprising way.
5. **Generate an angle between the topic and punch line.** The angle is a sentence or phrase that connects the topic to the punch line in a natural-sounding way.

This paper presents Witscript, a system for generating improvised joke responses in conversations that is inspired by those algorithms created by a professional comedy writer (Toplyn 2014, 2020a). Human evaluators judged Witscript's improvisations to be jokes more than 40% of the time, compared with less than 20% for the output of an advanced conversational response generation model.

### Related Work

The Witscript system generates jokes that depend on wordplay, the clever manipulation of the sounds and meanings of words. Amin and Burghardt (2020) provide a comprehensive overview of existing systems for the computational generation of verbal humor, including systems based on wordplay and puns. But almost all of those wordplay systems generate self-contained puns instead of the contextually integrated puns that would be more useful in practical applications (Ritchie 2005).

As far as I know, only the PUNDA Simple system (Dyballa et al. 2008) generates wordplay that is contextually integrated into a conversation as part of a response to a user’s utterance. However, that system generates a joke by extracting a noun from a user's utterance, retrieving a sentence containing a punning word from a database, and responding to the user with a part of the retrieved sentence that starts with the punning word. By contrast, the Witscript system generates a wordplay punch line based on two words or word chunks extracted from the user's utterance. Witscript then places that punch line at the end of its joke response, as part of a sentence that in most cases is generated on the spot and not retrieved from a database.

Google's neural conversational model Meena (Adwardana et al. 2020) has improvised at least one wordplay joke ("Horses go to Hayvard") in a conversation. But unlike Witscript, Meena has not been specifically designed to generate jokes. Therefore, the amount and timing of Meena’s joke output can't be controlled.

### Description of the Witscript System

Toplyn (2014) presents six algorithms, called Punch Line Makers, for creating a punch line. The Basic Joke-Writing Algorithm underlies several of those Punch Line Makers, including the one that Witscript currently employs to generate a joke response to a user’s utterance. Witscript uses Punch Line Maker #4: Find a play on words in the topic.

Wordplay can produce a humorous effect (Mihalcea and Strapparava 2005; Westbury and Hollis 2019). In terms of the Two-Stage Model (Suls 1972), Witscript uses wordplay to create an incongruous punch line that the user can make sense of by using a cognitive rule to connect it back to particular words in the user's utterance.

Because wordplay plays a central role in Witscript, first I'll discuss how the system calculates a wordplay score to measure the quality of the wordplay exhibited by two given words. Then I'll describe the rest of the system and how the wordplay score is used.

### Calculating the wordplay score

The Witscript system uses the pronunciation of words, i.e., their representation in phonemes, as given by the Carnegie Mellon University Pronouncing Dictionary (which is available at http://www.speech.cs.cmu.edu/cgi-bin/cmudict). The wordplay score for any pair of words is composed of these six subscores:
**Edit distance subscore**  This is the edit distance between either the pronunciations or, if a pronunciation is not in the dictionary, the spellings of the two words, as measured by the Levenshtein distance (Levenshtein 1966). Punning can include both phonetic and orthographical similarity (Valitutti, Doucet, Toivanen, and Toivonen 2015). The smaller the edit distance, the better the wordplay.

**Alliteration subscore**  This is 1 if the two words start with the same consonant phoneme. More points, which are constants, are added if more phonemes match. The greater this subscore, the better the wordplay. Alliteration and assonance tend to increase the funniness of a joke (Attardo, Attardo, Baltes, and Petray 1994; Mihalcea, Strapparava, and Pulman 2010).

**Assonance subscore**  This is a constant if the two words rhyme. If they don't, this subscore is the number of stressed vowels in one word that are repeated in the other word. The greater this subscore, the better the wordplay. The unexpected rhyming of two words that have different meanings can have a surprise effect (Attardo 1994), which, because the rhyming is harmless, results in funniness.

**Stop consonant subscore**  This is the total number of stop consonant phonemes (B, D, G, K, P, T) in both words. The greater this subscore, the better the wordplay. Stop consonants tend to make a joke funnier (Gultchin, Patterson, Baym, Swinger, and Kalai 2019; Tophlyn 2014).

**Ending subscore**  This is 1 if the last phoneme of both words matches. More points, which are constants, are added if more phonemes match at the ends of the words. The final syllable is particularly important in wordplay (Attardo 1994). The greater this subscore, the better the wordplay.

**Syllable subscore**  This is 1 if the two words have the same number of syllables, which makes their pairing more improbable, more surprising, and therefore funnier. To calculate the wordplay score for a pair of words, their six subscores are weighted and summed. The weights, along with the constants embedded in the subscores, were determined experimentally based on the quality of the system's output.

Now I'll describe Witscript as a whole and how it uses the wordplay score to create punch lines.

**Selecting two keywords from the topic**

The process of generating a joke response begins when the Witscript system receives a sentence from a user, which it treats as the topic of a possible joke. From that sentence, the system extracts the nouns, noun phrases, and named entities using the Natural Language Toolkit (available from https://www.nltk.org/).

Any extracted nouns, noun phrases, and named entities that are on a list of stop words are excluded from consideration. The stop words include the most commonly used words plus other words that, in my expert opinion, are unlikely to be useful for generating a funny joke, words such as "official," "person," and "tonight."

The remaining nouns, noun phrases, and named entities that have been extracted become the candidates for topic keywords because the humor of human-created jokes tends to be based on nouns and noun phrases (Dybala, Ptaszynski, and Sayama 2011; West and Horvitz 2019).

The topic keywords will serve as the topic handles described above in the Basic Joke-Writing Algorithm. Therefore, the system selects as the topic keywords those two topic keyword candidates that are the most likely to capture the user's attention. The two topic keyword candidates that are the most likely to capture the user's attention are assumed to be the two candidates that are the least likely to appear together in the topic sentence.

To select the two topic keyword candidates least likely to appear together in the topic sentence, the system uses word embeddings created by Word2Vec (Mikolov, Chen, Corrado, and Dean 2013). Using Word2Vec word embeddings for this purpose seems reasonable because if those embeddings indicate that two words are unlikely to appear together in a large, general, training corpus, then those two words are also unlikely to appear together in a topic sentence received from a generic user during open-domain chat.

The Gensim library (Rehurek and Sojka 2010) is used to load a pretrained Google Word2Vec model (Google-News-vectors-negative300.bin) which was trained on about 100 billion words of Google News data. Then the "similarity" function of the Gensim implementation of Word2Vec is used to select the two topic keyword candidates that have the least cosine similarity. Those candidates become the topic keywords. For example, when the user inputs, "I just read that some flower that smells like a corpse is about to bloom," the system selects as the topic keywords "flower" and "corpse."

Next the system tries to link those two selected topic keywords by means of wordplay to create an incongruous punch line. Three types of wordplay punch lines are attempted: juxtaposition, substitution, and portmanteau.

**Creating a juxtaposition punch line**

This type of punch line consists of two words right next to each other that, incongruously, exhibit wordplay. Juxtaposing two words can yield an incongruity that produces humor (Gultchin et al. 2019).

To create a juxtaposition punch line, the system starts by listing the top fifty words that are most closely associated with each of the two topic keywords; the number fifty was determined experimentally based on the quality of the system's output. To select those associated words, which I call associations, the system uses the "most similar" function of the Gensim implementation of the Google Word2Vec model trained on Google News data.

Next the system pairs the first topic keyword and each of its fifty associated words with the second topic keyword and each of its fifty associated words. The system selects the pair of words with the best wordplay score, one word from each list, to be the juxtaposition punch line. For the
example above, Witscript derives the juxtaposition punch line "garden carcass" from "flower" and "corpse."

Creating a substitution punch line

This type of punch line consists of a multi-word chunk into which a new word has been substituted, incongruously, for a word with which it has wordplay. This word substitution can produce a humorous effect (Binsted and Ritchie 1994; Valitutti et al. 2015).

The system creates a substitution punch line by pairing each topic keyword with each of the words in every multi-word chunk in the top fifty "most similar" associations of the other topic keyword. The system selects the pairing with the best wordplay score. Then it substitutes one word in that pair for the other in the relevant multi-word chunk to create the substitution punch line.

For example, when the user inputs, "People are trying to summon a Mexican demon by getting him to spin a pencil," Witscript derives the substitution punch line "Puerto Demon" from "Puerto Rican," which it associates with "Mexican," and "demon."

Creating a portmanteau punch line

This type of punch line entails the incongruous blending of two words into a portmanteau. The syllable substitution involved can result in humor (Binsted and Ritchie 1994).

The system pairs each topic keyword and its top fifty associations with the other topic keyword and finds a pair in which one word has a pronunciation that is similar to, but not identical to, the pronunciation of the beginning of the other word.

Then the system uses the Pyphen module (available from https://pyphen.org/) to divide the longer word into syllables, allowing the shorter word to be substituted for the equivalent number of syllables at the beginning of the longer word. This creates the portmanteau punch line.

For example, when the user inputs, "Researchers at Johns Hopkins have discovered a virus that causes stupidity," Witscript derives the portmanteau punch line "flupidity" from "flu," which it associates with "virus," and "stupidity."

Selecting the best punch line

The system attempts to generate one punch line candidate of each of the above three types. Each punch line candidate that is generated has a wordplay score. The wordplay score of a juxtaposition punch line candidate is the wordplay score that was calculated for the two words selected to be the juxtaposition punch line. The wordplay score of a substitution punch line candidate is the wordplay score of the words that were substituted for one another to create the substitution punch line. The wordplay score of a portmanteau punch line candidate is the wordplay score of the word and syllables that were substituted for one another to create the portmanteau punch line.

Whichever punch line candidate has the best wordplay score is selected for inclusion in the system's joke response. This is because the best wordplay score is a proxy for the biggest incongruity. The punch line that embodies the biggest incongruity is the most surprising. And the punch line that is the most surprising is most likely the funniest (Suls 1972; Toplyn 2014).

The system filters out any punch line candidate that has a wordplay score worse than a preset, empirically determined threshold. If the system hasn't generated any punch line candidate with a wordplay score better than or equal to the threshold, then it doesn't output any joke response.

Adding an angle to the selected punch line

After the system selects the best punch line, it adds an angle, which is text intended to smoothly connect the punch line to the user's input sentence.

To generate text to fill that gap, I used the language model BERT (Devlin, Chang, Lee, and Toutanova 2019) and the resources of Hugging Face, starting with their BERT, large, uncased model (available from https://huggingface.co/bert-large-uncased). That model had been pretrained for masked language modeling and next sentence prediction on BookCorpus, a dataset consisting of 11,038 unpublished books, and English Wikipedia (excluding lists, tables, and headers).

To fine-tune that pretrained BERT model, I used a dataset of late-night TV comedy show monologue jokes. To create that dataset, I first scraped 43,145 jokes from the archives available at these three websites: https://www.newsmix.com/jokes/archive/ https://github.com/brendansudol/conan-jokes-data http://www.101funjokes.com/political_jokes.htm

Next I prepared the dataset by taking the following steps: 1) I removed any duplicate jokes; 2) I added a topic to any joke that did not explicitly include one because, for example, the joke was a follow-up joke that implicitly assumed the topic of the previous joke; 3) I removed any extraneous words from the beginning and end of each joke like, for example, lead-in or concluding text that was only tangentially related to the actual joke; 4) I removed any joke that wasn't at least 17 tokens long because inspecting the dataset revealed that jokes shorter than that tended to be formed unclearly, lacking a distinct topic, angle, or punch line; 5) I removed any joke that was longer than 49 tokens because inspecting the dataset revealed that those longer jokes tended to be so wordy that the individual parts of the actual joke were hard for me to discern; 6) I manually annotated the remaining 36,781 jokes by splitting each of them into two segments—the topic and the rest of the joke—thus formatting them to use as input for fine-tuning the BERT model.

I used the prepared dataset of jokes to fine-tune the pretrained BERT model for one epoch and also for two epochs. Then I tested those two fine-tuned models, and the
pretrained BERT model without any fine-tuning, using as input the topics and punch lines of 48 monologue-style jokes I had written that were not part of the fine-tuning dataset. The BERT model that had been fine-tuned for one epoch generated the best angle, in my judgment, considerably more often than the other two models did. So the model fine-tuned for one epoch became the joke-tuned BERT model that the Witscript system uses.

Here’s how the Witscript system generates an angle using the joke-tuned BERT model. The BERT model is given the selected punch line, to which a [MASK] token has been prepended, together with the user’s input sentence. BERT predicts a token to fill the mask position and then prepends that token to the punch line. Predicting tokens backward from the punch line in this way seems empirically to produce more natural-sounding output than predicting forward from the topic. BERT continues prepending [MASK] tokens and predicting tokens until it generates a stop condition, such as a punctuation mark or a repeated token. Whatever text BERT generated before the stop condition becomes the angle, which is prepended to the previously selected punch line to form Witscript’s basic joke response.

Occasionally the very first token that BERT predicts is a punctuation mark, which indicates that BERT will be generating gibberish that won’t be useable as an angle. In that case, the system selects an angle template at random from a list I wrote and inserts the selected punch line into it to form Witscript’s basic joke response.

Finalizing the joke response

The Witscript system prepends to the basic joke response a filler word randomly selected from another list I wrote. Filler words such as "um" and "like" make the response sound more humanlike (Duvall, Robbins, Graham, and Divett 2014).

The system also repeats the two topic keywords at the beginning of its response, to help the user find the connections between the user’s input and the punch line; the system sometimes makes connections that are a bit obscure, and if a joke is too hard to process, it won’t be funny (Attardo et al. 1994).

Then the system outputs its final joke response to the user. Here’s how the system responds to the example user inputs above:

**Input:** "I just read that some flower that smells like a corpse is about to bloom."

**Witscript:** "Flower corpse? Heh, so now it smells like a garden carcass."

**Input:** "People are trying to summon a Mexican demon by getting him to spin a pencil."

**Witscript:** "Mexican demon? Mmm-hmm, or a Puerto Demon."

**Input:** "Researchers at Johns Hopkins have discovered a virus that causes stupidity."

**Witscript:** "Virus stupidity? Um, and not because of fluidity."

As an illustration of how Witscript produces its output, consider that last example:

- "Virus stupidity?" and the filler word "Um" were added when finalizing the joke response.
- The angle "and not because of" was generated by the joke-tuned BERT model.
- The punch line "fluidity" was created as described above in the section "Creating a portmanteau punch line" and selected as the best punch line.

System Evaluation

To evaluate the Witscript system fairly, I wanted to compare it to a strong baseline. It seemed to me that a strong baseline would not be a model that generates random responses to input sentences, but instead a model that generates joke responses that attend to the input sentences.

But to my knowledge no such input-attentive joke generators exist that work in English; the PUNDA Simple system (Dybala et al. 2008) generates joke responses in Japanese. So, without access to a demo of Google’s Meena (Adiwardana et al. 2020), I decided to use Microsoft’s neural conversational response generation model DialoGPT (Zhang et al. 2020) as a baseline model. Although DialoGPT was not designed to generate joke responses, at least it can generate a response that attends to an input sentence.

To evaluate Witscript, I selected 20 monologue-type jokes that professional comedy writer Joe Toplyn had written and posted on his Twitter account @JoeToplyn. None of those 20 selected jokes had been used to fine-tune BERT. The topic sentence of each selected Twitter joke became an input for testing the Witscript and DialoGPT systems. The part of each selected Twitter joke that came after its topic sentence became the gold-standard, human response to that input. That is, each selected Twitter joke provided an input for testing and also the human response to that input.

All of the 20 Twitter jokes selected to evaluate the system met the following criteria:

- Their punch lines, like Witscript's, featured wordplay. That requirement minimized the effect of any bias introduced by human evaluators who don't like wordplay jokes.
- Their topic sentences appeared to have at least two nouns, noun phrases, or named entities that were in the vocabulary of the Google Word2Vec model. That made it more likely that Witscript would output responses.
- Their topic sentences didn't include any named entities for which the Google Word2Vec model was likely to yield stale associations. That way, Witscript wouldn't be penalized for having to rely on a model trained on an old, static, news dataset.

I input the topic sentences of the 20 selected Twitter jokes into Witscript but only received responses from Witscript for 13 of those topic sentences. For the topic sentences of the other 7 selected Twitter jokes, Witscript was apparently unable to generate any punch line candidate that had a
wordplay score better than or equal to its internal threshold.

The 13 topic sentences for which Witscript did output responses were then used as input to obtain 13 responses from DialoGPT. To obtain responses from DialoGPT, I used Hugging Face's implementation of the model DialoGPT-large, which had been trained on dialogue from Reddit discussion threads (available from https://huggingface.co/Microsoft/DialoGPT-large). I started a new chat before inputting each topic sentence to DialoGPT, so as to eliminate any influence of dialogue history on the responses of DialoGPT.

I hired workers via Amazon Mechanical Turk to evaluate the responses generated by Witscript, DialoGPT, and the human for each of the 13 input sentences. The only qualifications that I specified for the workers were that they had to be located in the United States and have a Human Intelligence Task (HIT) Approval Rate greater than or equal to 95%. Therefore, nothing else is known about the evaluators.

The evaluators were not told how any of the responses had been obtained. The 39 input and response pairs were put in a random order and presented in that same order to all of the evaluators. 45 evaluators each read from 1 to 39 input and response pairs. Given each input and response pair, the evaluators were asked to rate the response by selecting either 1 (not a joke), 2 (almost a joke), 3 (a joke), or 4 (a very good joke).

I used that relatively simple 4-point scale because it directly measures the degree to which the evaluators perceive Witscript's responses to be jokes. If Witscript's responses are perceived to be jokes, then the system can be used to equip a conversational agent with humor, which would enhance the agent's performance (Dybala et al. 2008). And enhancing the performance of conversational agents such as chatbots was my main goal in creating Witscript.

Each input and response pair was evaluated by 15 workers and the mean of their ratings for the response in each pair was then calculated. Table 1 shows the first response that was output by the two systems for each input, together with the human response. All responses were transcribed exactly as they were output. The mean rating given by the evaluators for each response is also shown.

<table>
<thead>
<tr>
<th>Source</th>
<th>Input and responses</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Input:</td>
<td>A man pulled out a gun at a Popeyes restaurant when he heard</td>
<td>1.53</td>
</tr>
<tr>
<td>DiaLogPT</td>
<td>they had no more chicken sandwiches.</td>
<td></td>
</tr>
<tr>
<td>Witscript</td>
<td>Popeyes gun? Um, not a Bubba Gun.</td>
<td>2.33</td>
</tr>
<tr>
<td>Human</td>
<td>He also objected to any new laws on bun control.</td>
<td>3.00</td>
</tr>
<tr>
<td>2. Input:</td>
<td>America's national forests are being severely damaged by the</td>
<td>1.73</td>
</tr>
<tr>
<td>DiaLogPT</td>
<td>illegal growing of marijuana.</td>
<td></td>
</tr>
<tr>
<td>Witscript</td>
<td>America marijuana? Yah, so now he wants to be not in Canada</td>
<td>2.27</td>
</tr>
<tr>
<td>Human</td>
<td>Which explains the new Forest Service mascot, Tokey the Bear.</td>
<td>3.40</td>
</tr>
<tr>
<td>3. Input:</td>
<td>Oktoberfest, Adidas is releasing sneakers that repel beer and</td>
<td>1.80</td>
</tr>
<tr>
<td>DiaLogPT</td>
<td>vomit.</td>
<td></td>
</tr>
<tr>
<td>Witscript</td>
<td>Adidas vomit? Ah, I like the Puma poop.</td>
<td>2.87</td>
</tr>
<tr>
<td>Human</td>
<td>And they're changing their slogan to &quot;Just Spew It.&quot;</td>
<td>3.27</td>
</tr>
<tr>
<td>4. Input:</td>
<td>Two men were arrested in the theft of a hundred thousand</td>
<td>1.80</td>
</tr>
<tr>
<td>DiaLogPT</td>
<td>diapers.</td>
<td></td>
</tr>
<tr>
<td>Witscript</td>
<td>Theft diapers? Like, they stole stroller.</td>
<td>2.00</td>
</tr>
<tr>
<td>Human</td>
<td>They were charged with petty larceny.</td>
<td>3.33</td>
</tr>
<tr>
<td>5. Input:</td>
<td>It's the twenty-fifth anniversary of the Blue Man Group.</td>
<td>1.73</td>
</tr>
<tr>
<td>DiaLogPT</td>
<td>I was going to say the same thing.</td>
<td></td>
</tr>
<tr>
<td>Witscript</td>
<td>Blue anniversary? Heh, welcome to the Bluebliee.</td>
<td>2.87</td>
</tr>
<tr>
<td>Human</td>
<td>To celebrate, they're starting their own health plan: Blue Man</td>
<td>2.80</td>
</tr>
<tr>
<td></td>
<td>Blue Cross Blue Shield.</td>
<td></td>
</tr>
<tr>
<td>6. Input:</td>
<td>Thirteen TGI Fridays were selling cheap liquor in premium brand</td>
<td>1.33</td>
</tr>
<tr>
<td>DiaLogPT</td>
<td>bottles.</td>
<td></td>
</tr>
<tr>
<td>Witscript</td>
<td>Thirteen Fridays? Okay, but I prefer a nine night myself.</td>
<td>2.07</td>
</tr>
<tr>
<td>Human</td>
<td>The company claims it’s a new promotion: TGI Fraudasys.</td>
<td>3.00</td>
</tr>
<tr>
<td>7. Input:</td>
<td>At a zoo, the first rhinoceros ever conceived by artificial</td>
<td>1.60</td>
</tr>
<tr>
<td>DiaLogPT</td>
<td>insemination was born.</td>
<td></td>
</tr>
<tr>
<td>Witscript</td>
<td>Rhinoceros insemination? Mmm-hmm, welcome to the zooosemination.</td>
<td>2.13</td>
</tr>
<tr>
<td>Human</td>
<td>Usually rhinos are conceived using that dating site for large</td>
<td>2.93</td>
</tr>
<tr>
<td></td>
<td>mammals, Ashley Mastodon.</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Responses generated for each input, with the mean rating of each response on a scale from 1 (not a joke) to 4 (a very good joke).
8. Input: Dow Chemical is turning raw sewage into drinking water.

DialoGPT  I think you mean sewage.  
Witscript  Dow Chemical drinking? Yup, so not Dowing.  
Human  To make it more appealing, they’re calling it Pooland Spring.  

9. Input: A new study says drinking sugar-sweetened beverages increases the risk of death.

DialoGPT  I think you mean to say that drinking sugar beverages increases the risk of death. 
Witscript  Beverages risk? Huh, I have to focus on drinks risks. 
Human  That’s why they’re renaming that soft drink Coca-Coma.  

10. Input: In one of its restrooms, the Guggenheim Museum is installing a solid gold toilet.

DialoGPT  I think that’s a new one. 
Witscript  Gold toilet? Yep, to make a toilet pure. 
Human  It’s perfect for people who have just eaten 24 carrots.  

11. Input: A brewery apologized for a beer that shares the same name as a Hindu deity.

DialoGPT  I think you mean a Hindu deity apologized for a beer that shares the same name as a brewery. 
Witscript  Shares Hindu? Heh, no Sikh stock. 
Human  No word yet on their plans for their other beer, Buddha-weiser.  

12. Input: A truck was stolen that contained half a million dollars’ worth of handbags.

DialoGPT  I’m sure it was a truck full of handbags. 
Witscript  Truck handbags? Yep, of vansbags. 
Human  The thief was arrested and the handbags were charged as accessories.  

13. Input: A man was caught during a home burglary wearing only lingerie.

DialoGPT  I’m not sure if that’s a good thing or a bad thing. 
Witscript  Burglary lingerie? Yes, and the burglaries bikinis. 
Human  He was charged with pantry larceny.  

Table 1 (continued).

Table 2 compares the systems based on their ratings. The second column shows that Witscript’s responses were rated, on average, about halfway between those of the DialoGPT baseline and the human, a professional comedy writer.

The last column of Table 2 shows the percentage of responses that the evaluators rated as "a joke" or "a very good joke" (i.e., the number of ratings of 3 and 4 divided by the total number of ratings). Witscript’s responses were judged to be jokes 41.5% of the time, compared to only 17.9% of the time for the responses of DialoGPT. Witscript’s response to Input #5 was actually rated higher than the human’s response, which could be the first time ever that a machine defeated a human expert in a joke-writing challenge.

Some of the joke responses generated by Witscript don’t make total sense. For example, the wordplay connecting Input #2 in Table 1 to Witscript’s response is clear, but the logic isn’t. Despite their occasional gaps in logic, fully 79.0% of Witscript’s responses in Table 1 were rated by the evaluators as 2 (almost a joke) or higher. These impressive results may be partly due to the fact that each Witscript response has, at least, the form of a joke, complete with a punch line. Therefore, Witscript probably takes advantage of the "charity of interpretation" effect: evaluators may perceive that each of the well-formed linguistic containers offered up by Witscript is an even more meaningful joke than it actually is (Veale 2016). Still, if humans judge a Witscript response to be a joke, for whatever reasons, then the response is likely to function as a joke in a conversational context.

Those evaluation results, in connection with the research about joke-equipped chatbots cited in the Introduction, lead to the conclusion that users would perceive a chatbot equipped with a Witscript module to be more humanlike and likeable than one without it.

Table 2: Comparison of the systems based on their ratings

<table>
<thead>
<tr>
<th>System</th>
<th>Mean rating</th>
<th>% jokes (ratings of 3 or 4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DialoGPT</td>
<td>1.61</td>
<td>17.9%</td>
</tr>
<tr>
<td>Witscript</td>
<td>2.28</td>
<td>41.5%</td>
</tr>
<tr>
<td>Human</td>
<td>3.10</td>
<td>85.1%</td>
</tr>
</tbody>
</table>

Discussion

Computational Creativity

I believe that the Witscript system demonstrates strong computational creativity instead of mere generation because its output exhibits three characteristics: novelty, value, and intentionality (Veale and Pérez y Pérez 2020; Ventura 2016).

The system’s output has novelty because the contextually relevant joke that the system generates in response to a new input has almost certainly never been created before by it or by any other agent.
The system's output has value, as shown by the ratings given to its responses by human evaluators.

And the system produces that novel, valuable output with intensionality in several ways: It restricts its generation process by using domain knowledge about how a professionally-written joke is structured. It creates punch lines using pretrained word embeddings as a knowledge base for obtaining semantically related words. It completes jokes in an autonomous fashion by using a language model fine-tuned on an inspiring set consisting of professionally-written jokes. Finally, it employs a fitness function to rank the generated joke responses and intentionally filter out some that don’t meet a preset threshold of value.

Contributions

In addition to presenting an implementation of computational creativity that could make a chatbot more humanlike and likeable, this paper makes the following contributions:
1. It presents a novel system that can automatically improve contextually relevant wordplay jokes in a conversation.
2. It presents a novel method for measuring the quality of wordplay and automatically identifying some jokes that aren't funny enough to be delivered.
3. It demonstrates how computational humor can be implemented with a hybrid of neural networks and symbolic AI, where the symbolic AI incorporates expert knowledge of comedy domain rules and algorithms.
4. It presents an extensible framework for generating original jokes based on an input sentence. That framework mirrors the Basic Joke-Writing Algorithm described in the Introduction.

Future Work

The following work is needed to enable the Witscript system to execute the steps of the Basic Joke-Writing Algorithm more effectively and thereby output more sophisticated and funnier joke responses.

Selecting Topic Keywords To consistently select the topic keywords that are the most potentially useful for punch line generation, the Witscript system needs to incorporate better natural language processing tools.

For example, with better named entity recognition, Witscript might have identified “TGI Fridays” in Input #6 in Table 1 as the name of a restaurant chain instead of the plural of a weekday. With better part-of-speech tagging of Input #11, Witscript might have identified “shares” as a verb instead of a noun related to the stock market.

Also needed is a method for selecting the pair of topic keywords most likely to capture the user’s attention that is better than using vector-space distance. For example, a better method applied to Input #9 would have selected “beverages” and “death” instead of “beverages” and “risk.”

Generating Associations Research should be devoted to developing a more effective association engine to list associations, i.e., words and phrases related to a topic keyword. Sometimes Witscript will list words and word chunks whose relation to a topic keyword isn’t obvious, which can lead it to assemble a weak, puzzling punch line.

For example, to generate its response to Input #10 in Table 1, Witscript came up with the obscure association “karat pure” for “gold”—probably as used in phrases like “18 karat pure”—and then substituted “toilet” for “karat” to create the weak punch line “toilet pure.”

This limitation of Witscript’s Word2Vec-based association engine is partly due to the fact that word embedding models such as Word2Vec define relatedness as the extent to which two words appear in similar contexts, which can be different from how closely associated two words are in the minds of humans (Cattle and Ma 2017). So future research might explore developing a more effective association engine that is a hybrid of a text-based external language approach and a word-association-based internal language approach (Deyne, Perfors, and Navarro 2016).

A more effective association engine to use in generating conversational jokes would also not be static, as Witscript’s Word2Vec-based engine is, but instead would be regularly updated. Updating the association engine regularly would not only increase the size of its vocabulary but also ensure that its associations are capturing fresh topical relationships, such as current events and what most people think about them (Cattle and Ma 2017). To do a better job of simulating a witty human companion, Witscript needs an association engine that can accurately answer a question like “What do most people today think of when they hear the words [name of a celebrity]?”

Generating an Angle Future work might improve the method that Witscript uses to generate the angles for its joke responses. The joke-tuned BERT model tends to generate angles that are fairly simple and not as specific to the input context as the human-written angles are.

But the BERT-generated angles do have the virtue of being short, which is good from a humor perspective: the shorter a joke is, the funnier it tends to be (Toplyn 2014). And the BERT model is capable of connecting a topic sentence to even the strangest punch line in a way that makes the system's output sound reasonably natural

For example, in contributing to the responses for Inputs #5 and #7 in Table 1, the BERT model apparently decided that the unique portmanteaus "Bluebilee" and "zoosmination" could be names for the noteworthy occasions described by the inputs. So for an angle, the BERT model supplied a logical way to introduce a noteworthy occasion: "Welcome to the..." Only for Input #6 in Table 1 did the BERT model fail to generate useable text, which led to the system turning to a prewritten template to complete its response.

Creating Punch Lines Currently the Witscript system creates punch lines that rely on wordplay. But as advances continue to be made in AI, this system could be extended to create punch lines by using Punch Line Makers that don’t rely on wordplay (Toplyn 2014). Such Punch Line Makers include techniques that rely instead on common-
sense knowledge to generate associations and to link them to create a punch line.

Consider this human-written example, adapted from a joke posted on the Twitter account @JoeToplyn:

Input: The U.S. is planning to buy 22 aging fighter jets from Switzerland.

Response: Yeah, the Swiss fighter jets have air-to-air missiles, smart bombs, a can opener, a nail file, and a toothpick.

If the Witscript system were equipped with common-sense knowledge, it might generate a joke response like the one in that human-written example by taking these steps:

1. Select as the topic keywords "fighter jets" and "Switzerland."
2. Determine that air-to-air missiles and smart bombs are parts of fighter jets.
3. Determine that a Swiss Army knife is related to Switzerland. Also determine that a can opener, a nail file, and a toothpick are parts of a Swiss Army knife.
4. Create a punch line that links the topic keywords "fighter jets" and "Switzerland" by generating this surprising juxtaposition of associations: "air-to-air missiles, smart bombs, a can opener, a nail file, and a toothpick."
5. Recall that the punch line is a list of parts of the entities specified by the topic keywords "fighter jets" and "Switzerland." Blend those topic keywords into the phrase "the Swiss fighter jets" so as to parallel the association "a Swiss Army knife." Use that "parts of" relationship to generate an angle by appending to "the Swiss fighter jets" the verb "have."
6. Concatenate a filler word, the angle, and the punch line to get the final joke response: "Yeah, the Swiss fighter jets have air-to-air missiles, smart bombs, a can opener, a nail file, and a toothpick."

Selecting the Best Joke The current implementation of the Witscript system determines which joke response to output by selecting the punch line candidate that incorporates the best-scoring wordplay. But future implementations of the system could employ a more comprehensive funniness score to determine which joke response candidate to output. The funniness score of a joke response candidate could comprise several feature scores, which would be weighted and combined to yield the funniness score.

Those feature scores could comprise the wordplay score of the joke's punch line and also an interest score measuring the degree to which the two topic keywords used to create the punch line are related to each other. The less related that the topic keywords are to each other, the more attention-getting they may be when appearing together in the topic, the better the interest score, and the better the funniness score (Toplyn 2020b).

The feature scores could also include a clarity score measuring the degree to which the two topic keywords are related to their respective associations that were linked to create the joke's punch line. The more closely related each topic keyword is to its association that was used to create the joke's punch line, the more understandable the joke, the better the clarity score, and the better the funniness score (Toplyn 2020b).

Conclusion

In this paper we have introduced Witscript, a novel joke generating system that can improvise conversational joke responses that depend on wordplay. The Witscript system, because it embodies the Basic Joke-Writing Algorithm, also provides a road map to generating more sophisticated joke responses that depend on common-sense knowledge.

But because the Witscript system in its current implementation seems to regularly generate acceptable joke responses, it could even now be integrated as a humor module into an open-domain chatbot (Sjobergh and Araki 2009). The proprietary software is available for license.

People often use chatbots to fulfill a desire for entertainment or socializing (Brandtzæg and Følstad 2017). So an open-domain chatbot that uses Witscript to occasionally ad-lib a pretty good joke might potentially animate an artificial, but likeable, companion for lonely humans.

References


Brandtzæg, P.B., and Følstad, A. 2017. Why People Use Chatbots. INSCI.


Mikolov, T.; Chen, K.; Corrado, G.S.; and Dean, J. 2013. Efficient Estimation of Word Representations in Vector Space. ICLR.


Toplyn, J. 2014. Comedy Writing for Late-Night TV: How to Write Monologue Jokes, Desk Pieces, Sketches, Parodies, Audience Pieces, Remotes, and Other Short-Form Comedy. Rye, New York: Twenty Lane Media, LLC.


West, R., and Horvitz, E. 2019. Reverse-Engineering Satire, or "Paper on Computational Humor Accepted Despite Making Serious Advances". AAAI.


Structures in Tropes Networks: Toward a Formal Story Grammar

Jean-Péïc Chou  
École polytechnique  
Palaiseau FRANCE  
jean-pei.chou@polytechnique.edu

Marc Christie  
Univ Rennes, Inria, CNRS, IRISA, M2S  
Rennes FRANCE  
marc.christie@irisa.fr

Abstract
Tropes are cultural narrative conventions that shape our expectations of stories. This paper proposes a new approach to examine movie content relying on tropes. It first presents the architecture of tropes ontology extracted from tvtropes.org, then studies the link prediction problem in trope bipartite networks and discusses the next challenges and numerous applications that ensue from this approach. In addition, we propose to assess the potential of tropes to be the lexicon units of a formal story grammar.

Introduction
Stories are the pillars of all human cultures. They shape our imagination, forge our understanding of the world, and ensure our cultural heritage. Stories have been particularly broken down and analyzed since early XXth century Russian formalism. Propp observed Russian tales regularity and identified 31 narrative functions and 7 types of characters (Propp 1968). Greimas later proposed the first role-based structure for tales with the actantial model and introduced the isotopy as the redundancy of semantical elements in a text ensuring the coherence of a story (A.J. Greimas 1966). Spearhead of the structuralism, Barthes proposed an exhaustive analysis model by decomposing stories into complex hierarchical structures of discontinuous and heterogeneous elements with interacting isotopies, understood as meaning units. He distinguished the Function (causality, chronology...), the Action (communication, desire, hardship), and the Narration (R. Barthes 1966), and suggested five conventional ways to interpret a text segment that he named codes (Barthes and Balzac 1970). This narratological heritage has largely been expanded by derived models since then.

Propp’s works notably inspired Lakoff who developed the first story grammar (G. Lakoff 1972). This milestone was followed by Colby’s specialized grammar of Eskimos folktales (Colby 2009) and the first general grammar by Rumelhart (David E. Rumelhart 1975) which could be applied to wider sets of stories. These grammars define rules that for instance break down stories into setting, theme, plot, and resolution, each of them being decomposed into other sub-elements such as characters, goal, or episode. However, these grammars have not been able to capture all stories specificities. Indeed, stories evolve with their tellers’ fantasies, as their structures, their forms, and their content do. In their evaluation of story grammars, Black and Wilensky pointed out the limits of such structures for story comprehension and urged instead to study the knowledge enabling to understand story content (John B. Black and Robert Wilensky 1979), Garnham also discredited story grammars invoking the finitude of lexicon story elements (Alan Garnham 1983). Eventually, although such studies have been exploited in a few computational contexts such as story generation, the high abstraction of these grammars or their lack of data have made them unsuited to computational uses.

To overcome these deficiencies, we propose to use tropes as interconnected elements of the lexicon of stories, structured in semantic networks. As part of this work, we present a thorough analysis of tropes in five main sections. Our first inquiries will be to introduce tropes definition and state-of-the-art in the first two sections. In the third section, we will dwell on our data itself, examine its structure, and propose a terminology to concepts revolving around. These grounds will enable us to study trope networks to extract statistical patterns suggesting underlying grammar rules in the fourth section. We will eventually conclude on the perspectives raised by this work.

The work presented here is part of a larger research effort to assess the potential of tropes to be the semantic unit of a formal language of stories.

Tropes Definition
Tropes are storytelling devices or conventions. They are recognizable patterns found in all kinds of media. Authors and creators use them as narrative tools to make a specific impression on the audience. Tropes can describe every level of a work: the story and its discourse, characters and their interaction, location, time... The Save the Princess trope depicts the universal story plot in which a character, often portrayed by the Damsel in Distress trope, is kidnapped, and later rescued by the Hero for instance. At a finer scale, tropes can also simply correspond to a way a scene is filmed by a camera. The Revealing Hug trope depicts how the camera sometimes focuses on a character whose eyes are open while hugging to suggest that something is on his/her mind. As tropes describe both stories and discourses, we can fully
break down and analyze narratives through the prism of tropes.

Tropes are inventoried on a website named *tvtropes* taking the form of a wiki. An active community of contributors have been adding tropes and artworks of all kinds (from advertisements to theater pieces) on the website since 2011. These data are obviously subjective and are therefore regularly modified. Some tropes are redefined, renamed, removed, while others have not been listed yet or emerge from the latest original movies. As such, tropes library is organic, dynamic, and extensive (García-Ortega et al. 2020b). It adapts to current events and will never be complete, evolving over the course of the debates on the forum. Tropes library is for this reason the most extensive story elements lexicon created.

Besides, tropes find their essence in people’s story imagination and are thus deeply rooted in one’s culture. Many branches of the wiki exist in different languages, containing culturally specific tropes. We will use the English version of the database for it is the most complete one and for the British-American culture is widely spread in the occidental world.

**Related Work**

As of today, only a few publications focus on or mention tropes. Among them, García-Ortega et al. propose an initial analysis of tropes in films highlighting key figures about *tvtropes* data (García-Ortega et al. 2020a). In his work on *tvtropes* website pages network, Meeks studies tropes main topics and manages to detect six weak communities that tend to settle along six themes (Meeks 2011). He then uses topic modelling to depict works from this classification. The nature and extend of these themes are however questionable as they seem vague and are centered around overly used, and not necessarily significant, tropes.

Tropes have otherwise mainly been used for story generation. Thompson assesses that tropes are well-suited for being story components of advanced grammar in such a task (Thompson 2018) as they are reusable, present a natural semantic hierarchy and can be combined to form more complex and nuanced components than plain Action Hero. He also states that a major advantage of tropes is their popular and cultural topicality, unlike unfriendly Propp’s concepts. He created *Tropical*, a trope-based programming language for story authoring which enables to combine and create tropes as building blocks of simple stories. Guarneri et al. (Andrea Guarneri et al. 2017) selected 94 tropes that they classified into narrative types and used them as story components for their game plot creation tool GHOST. Tropes library was completed with cards from the creative library of James Harris’ way to picture tropes roles in stories structure (Harris 2017). In his artwork named *The Periodic Table of Storytelling*, he classifies tropes based on their Narrative category to build a periodic table made of story atoms. Eventually, García-Ortega et al. built an evolutionary algorithm that generates original sequences of tropes coupled with a neural network predicting movies success from their tropes (García-Ortega et al. 2020b). However, the generated lists lack chronological order and more generally interpretability.

Tropes have also been mentioned in movie genre classification tasks. In their study in trailer generation, Smith et al mention weak tropes as visual elements characterizing genres (Smith et al. 2017). They note statistical patterns between colors, objects or places and genres: astronauts are present in drama movies while waitresses are found in romances or comedies. Äijälä looks for tropes similarities between movies and assesses the results with a genre analysis (Äijälä 2020). Results on most popular movies are encouraging at a small scale but the hierarchical clustering fails at forming coherent larger communities.

Eventually, *tvtropes* data has been used to conduct diverse cultural studies such as Mellina and Svetlichnaya’s analysis of the evolution of the bipartite network with widely used emerging and dying tropes (Mellina and Svetlichnaya, 2011). Otherwise, gender bias in tropes is studied by Gala et al. (Gala et al. 2020), and by Assogba et al. from Bocoup company with an interactive visualization of tropes (Assogba etal.).

**Tvtropes Data Networks**

In traditional grammar, nature and function are used to define the type and the role of a word in a sentence. Knowing them notably allows us to apply the correct grammar rules. In this section, we will aim at structuring our heterogeneous data to have a sense of the nature and the function of tropes. We will study the *tvtropes* resource as an ontology by extracting and defining underlying networks from the website and more concretely, explore the themes covered by the data.

**Tropes Data**

*tvtropes* is divided into two main parts. The first main part of *tvtropes* includes tropes information. Each of the tropes is always provided with a thorough description which can include its definition, a stereotypical example, the context in which the trope often appears, its supposed origins and its evolution, or the tropes it is close or antinomic to with the mention See also. Below, we can generally find an exhaustive list of examples and their context in various media. The description and the examples can also be accompanied with:

- A **Laconic** page which gives a short definition of the trope,
- A **Quotes** page presenting extracts from works that deal with the trope or give an example of it,
- A **Playing With** page listing several examples of variations as described above,
- An image illustrating the point,
- A video example.

More than 27,000 tropes have been identified, some having been used in more than 2000 different works. A case in point is the most used trope *Shout Out*, which describes a
work referring a person or another work. Figure 1 shows the repartition of the number of listed uses in movies by trope.

**Figure 1:** Repartition of tropes based on their appearance

### Media and the Bipartite Networks

The second main part of the wiki focuses on Media, which correspond to artworks of all sorts. Each work is described similarly to Wikipedia pages and can include links to other pages giving anecdotes, iconic moments, common thoughts, summaries, other information. More interestingly, a list of characters with tropes describing them is sometimes available.

Tropes have been listed for each work and each character. The more fans the movie has, the more complete the inventory of tropes is (García-Ortega et al. 2020b). Figure 2 presents the repartition of the number of listed tropes by movie. We observe that some films include more than 600 tropes which corresponds to 1 trope every 12 seconds on average for a 2-hour-long movie. Most described movies are super-hero blockbuster movies (Thor Ragnarok, Thor, X-Men Apocalypse…).

**Figure 2:** Repartition of movies based on their number of listed tropes

The website structure naturally forms a bipartite graph linking tropes to artworks or characters. Whether focusing on works or characters, we will call this structure the bipartite network. The projection of the bipartite network results in a co-occurrence graph. These co-occurrence graphs are particularly interesting since they allow to link movies with similar tropes, i.e. similar plot events, for instance.

### The Structural Networks

**Tvtropes Classification**

Tvtropes divides tropes into multiple hierarchical categories and subcategories as on Wikipedia. Tropes are mainly classified into four overlapping folders. The names of the categories explicitly describe the kind of classification inside each of them:

- **Genre** (Action/Adventure, Comedy, Horror…)
- **Media** (Film, Game, Theatre…)
- **Narrative** (Characters, Plots, Symbols …)
- **Topical** (Alien, Betrayal, Combat …)

Most of the time, tropes belong to several categories or sub-categories. There are no clear levels of folders: a folder can be at two levels below and one level below a category at the same time as shown in Figure 3 for instance. There can namely be multiple paths of varying distances from a folder to another in this data structure. We will call the network resulting from this hierarchy and categorization, the structural network.

**Figure 3:** Example of a hierarchical overlapping found on tvtropes.org

**Narrative subcategories and Abstraction Scale**

We observe that the narrative subcategories describe functions of tropes in a story, some of them having a more or less significant impact on it. For example, plots tropes will naturally be more relevant to the story than Symbol ones. From this intuition, we propose to determine an abstraction scale of tropes, namely defining levels of description. For this, we exploit the proposed Narrative subcategories which include:

- **Applied Phlebotinum**: Unexplainable plot fuel, something that enables scenario shortcuts (e.g. futuristic technology, magic and powers, tools…)
- **Characterization**: Describe characters, his/her origins, appearance, moral values…
- **Characters**
- **Characters & Casting**: Impact of human actors playing roles
- **Characters as Device**: How a character serves a plot
- **Character Introduction Index**: How characters are introduced
- **Conflict**: Problems that drive a story
- **Dialogue**
- **Genre** (main folder)
- **Motifs**: Symbols that recurrently appear in a work
- **Narrative Devices**: What moves the story forward or organize a scene or sequence
- **Narrator Tropes**
• **Paratext:** Not directly contained in the work content (DVD box, preface…)
• **The Plot Demanded This Index:** Something happening for no other reason than ‘the plot requires it’
• **Plots:** Organize the action of an entire script or episode
• **Settings:** Time, location, and circumstances of the narrative
• **Show, Don’t Tell:** Demonstrative techniques rather than informative ones
• **Spectacle:** Techniques to impress the audience (lighting, props, camera tricks…)
• **Symbolism:** Common representation of an idea, belief, event…

Some of these folders contain or belong to each other. We arbitrarily decide not to consider *Characters & Casting, Characters as Device, Conflict, and Genre* to limit the overlapping and to work with heterogeneous categories. These categories correspond to tropes functions in stories.

We pinpoint some strong correlations between categories which have tropes in common. We first consider *Plots* tropes as the highest level descriptors of stories, and thus place them at the top of our abstraction scale. We then naturally build a scale by placing alongside correlated categories and obtain a result observable in Figure 4. We note that:

• **Paratext**, **Settings** and **Applied Phlebotinum** are put aside as they are independent categories, uncorrelated to other ones.
• The **Dialogue** tropes can both be placed at a high and low level. Dialogues can reveal plot-twist and be at the center of a story while many of them only fill a work.

<table>
<thead>
<tr>
<th>Abstraction</th>
<th>How?</th>
<th>In which context?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paratext</td>
<td>Settings</td>
<td>Applied Phlebotinum</td>
</tr>
<tr>
<td>Characters</td>
<td>Narrative</td>
<td>Devices</td>
</tr>
<tr>
<td></td>
<td>Dialogue</td>
<td>The Plot Demanded</td>
</tr>
<tr>
<td>Characters</td>
<td>Char. Introduction</td>
<td>Characterization</td>
</tr>
<tr>
<td>Characters</td>
<td>The Plot Demanded</td>
<td>Spectacle Motifs</td>
</tr>
<tr>
<td>Characters</td>
<td>The Plot Demanded</td>
<td>Symbolism</td>
</tr>
<tr>
<td>Characters</td>
<td>The Plot Demanded</td>
<td>Show Don’t Tell Me</td>
</tr>
</tbody>
</table>

![Figure 4: Proposed abstraction scale built from narrative categories correlations](Image)

**Semantic networks**

From the wiki, we can extract three semantic networks that connect tropes with semantic relations:

• **Synonymy:** *The Revealing Hug* trope is very similar to the *Traitor Shot* trope as they both describe ways to film the suspect behavior of a character.
• **Antonymy:** *Heroes* versus *Villains*.
• **Metonymy:** *Revealing Hug* is considered as a sub-trope of *Meaningful Look*.
• **Lexical field:** *Save the Princess Plots* is most of the time associated with the *Damsel Distress Characters* trope for instance.
• **Causality and dependency**

We will call these structures as follows:

• **Description** network: Tropes’ descriptions mention other tropes, pointing to each other via hyperlinks.
• **Related** network: Connections are also extracted from the Related page of each trope. It lists all the pages mentioning at least once the trope. This method completes the previous one. In addition to the descriptions, links can be found in the examples, captions… Both networks correspond to subparts of the tvtropes pages network analyzed by Elijah Meeks in 2011 (Meeks 2011).
• **Example** network: We note that examples of use of tropes on movie pages can refer to other tropes when providing the context. If the explication needs to include another trope to tell the context, it means that they share some semantic relation.

**Datasets**

How was this data collected? Tropes data were first extracted and made available for the scientific community by Kiesel and Grimmes with an RDF-dataset (M. Kiesel, G. A. Grimmes 2010). The last downloadable dump was generated 1st of July 2016 and contains over 20 million RDF statements for a total size of 4.7 GB. However, the size of tvtropes database has since then tripled. García-Ortega et al. released *PicTropes*, a more recent dataset including 5,925 films and 18,270 tropes in 2018 (García-Ortega et al. 2018). In this regard, they designed a downloadable web crawler named *tropescraper*. It first extracts all the movies from tvtropes then seeks for all the listed tropes on each film page.

We modified and made use of *tropescraper* to extract other available data previously mentioned (tropes categories, tropes per character, semantic networks links).

**Exploring Topics in Tropes**

These structures have given us a sense of the type of information tropes describe. But in more concrete terms, what are the topics covered by tropes and what are they about? Following Meeks’ study (Meeks 2011), we worked on the network of tropes pages. Our attempts to obtain similar themes with community detection were not conclusive. This can be explained by the considerable growth of tvtropes website since 2011. Instead, we extracted tropes topics by studying tropes descriptions and examples with Latent Dirichlet Allocation (LDA) model and topic modelling. We used Genlin library implementation of LDA and define the number of topics by computing the topic coherence score for several numbers.

An initial overview of tropes descriptions presents overused words (e.g. “character”, “one”, “compare”, “work”). We chose to filter words appearing in more than one document out of two. We ran a first analysis on tropes descriptions for which we find the best coherence with 18 topics. The intertopic distance map and three topic descriptions are shown Figure 7. Topic 1 clearly relates to the work as an entity and topic 3 to the characters used to describe a situation. Topic 4 refers to abilities and is close to video games lingo. Topic 3 is harder to semantically distinguish. It mixes notions of family, love, evil, or fuzzy words like car.
Figure 7: Intertopic Distance Map and descriptions of topics 1, 3, 4 from tropes descriptions with LDA model

Figure 8: Intertopic Distance Map and descriptions of topics 1, 2 and 3 from tropes descriptions and examples with LDA model
We conducted the same study with tropes descriptions and their examples. We supposed here that the vocabulary employed is representative of each trope. Coherence graph indicated a local maximum around 26 topics. Extracted topics found in Figure 8 illustrate well the diversity of the themes. From love in topic 2 to weapons and video games in topic 3, or topic 1 and suffering.

This study on tropes descriptions shows the diversity of addressed topics and confirms the plurality of the abstraction levels scanned. In addition, it demonstrates the existence of main themes for future topic modelling analysis.

**Link Prediction in the Bipartite Network**

We will now study tropes within their networks. Predicting tropes presence in a movie from given tropes and information about the movie would suggest the existence of statistical patterns and a semantic logic in tropes interactions. In other words, this would confirm that there is an ensemble of latent rules to follow in order to understand tropes language. This task is equivalent to the well-known link prediction problem.

**Problem Definition & Initial Datasets**

Our first objective is thus to build an algorithm inferring removed links between movies and tropes of the bipartite network. Removed connections depend on our goal. We create and name initial datasets as follows:

- **80-20**: For each movie, we remove 20% of the links with tropes to appraise the overall performance.
- **N-random**: We separate the movies in two parts. The first group (80%) remains intact and serves as training, while the second is connected to a chosen batch of N randomly chosen tropes (20%), serving as test. This situation would happen if we concentrated our efforts in conceiving algorithms dedicated to detecting N tropes.
- **Narrative category**: Following the previous method, we prepare train-test datasets by removing all links to tropes not in the studied narrative category.

We arbitrarily restrain our study to movies containing more than 5 tropes. This corresponds to 95% of the dataset. Tropes that have been used less than 5 times are also removed. The graph is an undirected bipartite multi-layer graph which can be weighted. Its main structure is given by the movies-tropes bipartite network. The other layers contain information about tropes from the previously built ontology, and information about movies content or metadata.

**Method**

We separately work with each layer of information to assess their efficiency in this task. We only make use of information that a naive algorithm detecting tropes would be able to obtain. Movie tags or collection are for instance not included. The graph is presented in Figure 9 for more clarity. The available information is used as follows:

- **Movies bipartite network**: We draw inspiration from Kunegis et al. who adapted link prediction functions based on probabilities of paths to bipartite networks by keeping the odd components (Kunegis et al.). We use this way odd polynomial link prediction functions, namely the Hyperbolic Sine and the Odd von Neumann Pseudokernel of the transition probability matrix with an empirical $\alpha = 0.1$.

\[
\sinh(\alpha, A) = \sum_{i=0}^{\infty} \frac{\alpha^{1+2i}}{(1+2i)!} A^{1+2i}
\]

\[
K_{\text{NEU}}^{\alpha}(A) = \alpha A(I - \alpha^2 A^2)^{-1} = \sum_{i=0}^{\infty} \frac{\alpha^{1+2i}}{(1+2i)!} A^{1+2i}
\]

- **Characters bipartite network**: We project the characters bipartite network on tropes nodes with edges weights given by the number of common neighbors. We compute the probabilities of paths between movies and tropes of a random walker starting on a movie, taking a first step to reach the tropes side, then taking a second on this resulting projected graph.

- **Semantic and structural networks**: For each network, we apply a similar method (the second step is made in the considered network). The networks are not weighted.

- **Movies metadata (genres, director, actors, released year)**: We use a similar method. This time, the random walker takes its first step from a movie to another one with probabilities relying on metadata similarities. Similarities are computed with the Jaccard method.

![Figure 9: Simplified representation of the multi-layer bipartite network](image)

For each dataset, we compute the probabilities of all tropes to be in a movie and rank them by order of likelihood. To evaluate the performance, we use area under precision-recall curve (AUPR) and area under ROC curve (AUC) as the classes are highly imbalanced (ratio of true edges in the order of 0.001). The metrics can be understood as follows: picking a random true removed link and a random false link, it is the probability that the computed likelihood of the true connection is higher than the false one.

**Results**

**80-20 Dataset** We report the results for the 80-20 dataset in Figure 10. Observations and analysis on this figure are listed as follows:

- For every dataset, best results are obtained from movies bipartite network and movies genres with respective AUC of 0.855 and 0.847. Both information sources rely on movies content similarities. It is reasonable to state
that two movies of the same genres or with similar tropes will have relatively close list of tropes. For further analysis on movie content, we could compare subtitles with LDA as Bougiatiotis and Giannakopoulos do (Bougiatiotis and Giannakopoulos 2016).

- **Characters** tropes also show helpful statistical patterns leading to an AUC of 0.745.
- Tropes categorization into genres and topics does not seem to be efficient in this task as we obtain 0.601 and 0.607 of AUC for the 80-20 dataset.
- Semantic networks are more useful. Especially the related-based one which reaches a score of 0.775 compared to a score of 0.653 for the description-based one. This result seems fair as the related network is denser.
- With AUC of 0.528 and 0.589, movies director and cast are not valuable, while surprisingly, movies released year gives a relatively better AUC of 0.648.

To assess the potential of a fine-tuned link prediction algorithm, we sum the computed probabilities of each of these layers by attributing to them an empirical weight. The weights reflect the previous results. They can be found in Figure 10. We obtain an AUC of 0.88 for the 80-20 dataset which corresponds to an improvement of 2.9% compared to the previous best score.

<table>
<thead>
<tr>
<th>Information source</th>
<th>Applied Philodendron</th>
<th>Characterization</th>
<th>Characters</th>
<th>Character Introduction</th>
<th>Dialogue</th>
<th>Meta</th>
<th>Narrative Devices</th>
<th>Narrator</th>
<th>Pastoral</th>
<th>The Plot Devices that Device</th>
<th>Plots</th>
<th>Settings</th>
<th>Show Don’t Tell</th>
<th>Spectacle</th>
<th>Symbolism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of tropes</td>
<td>1066</td>
<td>7146</td>
<td>4739</td>
<td>53</td>
<td>4739</td>
<td>68</td>
<td>5461</td>
<td>276</td>
<td>303</td>
<td>334</td>
<td>2412</td>
<td>2210</td>
<td>71</td>
<td>2860</td>
<td>55</td>
</tr>
<tr>
<td>Bipartite (sinh)</td>
<td>0.607</td>
<td>0.803</td>
<td>0.811</td>
<td>0.559</td>
<td>0.759</td>
<td>0.532</td>
<td>0.806</td>
<td>0.609</td>
<td>0.642</td>
<td>0.668</td>
<td>0.799</td>
<td>0.735</td>
<td>0.566</td>
<td>0.779</td>
<td>0.533</td>
</tr>
<tr>
<td>Bipartite (neumann)</td>
<td>0.607</td>
<td>0.803</td>
<td>0.812</td>
<td>0.559</td>
<td>0.759</td>
<td>0.532</td>
<td>0.806</td>
<td>0.609</td>
<td>0.643</td>
<td>0.668</td>
<td>0.800</td>
<td>0.735</td>
<td>0.566</td>
<td>0.771</td>
<td>0.533</td>
</tr>
<tr>
<td>Characters</td>
<td>0.535</td>
<td>0.671</td>
<td>0.685</td>
<td>0.530</td>
<td>0.657</td>
<td>0.514</td>
<td>0.704</td>
<td>0.540</td>
<td>0.532</td>
<td>0.590</td>
<td>0.678</td>
<td>0.591</td>
<td>0.521</td>
<td>0.646</td>
<td>0.513</td>
</tr>
<tr>
<td>Description Semantic</td>
<td>0.497</td>
<td>0.560</td>
<td>0.543</td>
<td>0.498</td>
<td>0.513</td>
<td>0.495</td>
<td>0.560</td>
<td>0.499</td>
<td>0.497</td>
<td>0.502</td>
<td>0.502</td>
<td>0.496</td>
<td>0.506</td>
<td>0.497</td>
<td>0.533</td>
</tr>
<tr>
<td>Related Semantic</td>
<td>0.513</td>
<td>0.656</td>
<td>0.636</td>
<td>0.503</td>
<td>0.568</td>
<td>0.496</td>
<td>0.663</td>
<td>0.510</td>
<td>0.503</td>
<td>0.526</td>
<td>0.593</td>
<td>0.546</td>
<td>0.551</td>
<td>0.559</td>
<td>0.499</td>
</tr>
<tr>
<td>Tropes Genres</td>
<td>0.508</td>
<td>0.564</td>
<td>0.566</td>
<td>0.503</td>
<td>0.540</td>
<td>0.496</td>
<td>0.575</td>
<td>0.507</td>
<td>0.499</td>
<td>0.528</td>
<td>0.563</td>
<td>0.533</td>
<td>0.500</td>
<td>0.545</td>
<td>0.498</td>
</tr>
<tr>
<td>Tropes Topics</td>
<td>0.500</td>
<td>0.496</td>
<td>0.544</td>
<td>0.498</td>
<td>0.485</td>
<td>0.495</td>
<td>0.539</td>
<td>0.503</td>
<td>0.496</td>
<td>0.510</td>
<td>0.536</td>
<td>0.513</td>
<td>0.496</td>
<td>0.512</td>
<td>0.497</td>
</tr>
</tbody>
</table>

**Figure 10:** Results of the link prediction for the 80-20 dataset and empirical weights of the sources of information for link prediction

**Figure 11:** Results of the link prediction by narrative category, best score for each source of information is shown in bold

**N-random & Narrative category** The study by narrative category and with N-random datasets is shown in Figures 11 and 12. Best score for each source of information is shown in bold. This analysis is not meant to prove the absolute efficiency of a category for this task but discriminates between the best and worst performing categories:

- **Characterization, characters, and narrative devices** tropes lead to the best results. However, we cannot conclude on the high informative value of these categories as these categories include the most tropes with 7146, 4739 and 5461 ones.
- It can still be noted that we obtain better results with characters (4739) than dialogue tropes (4739), and with plots (2412) than settings (2210) as these categories include close number of tropes. This result confirms our initial intuition about high-level tropes.
- N-random datasets present higher AUC scores than their narrative counterparts with similar number of tropes. This indicates that heterogeneous information is more beneficial to predict links.

**‘Small’ Movies and Characters**

These results must be examined cautiously. Since tvropes dataset is inconsistent and biased, even a human would not be able to guess which tropes have been inventoried or not. The community might have based their list on a single scene or an insignificant detail. Besides, fully described movies such as Star Wars include more than 600 tropes which cannot make sense without indication of context.

However, we conducted the same work on characters and movies containing between 5 and 30 tropes. With AUCs of 94 and 91, we note a clear improvement that indicates that these works are more easily understandable with our data and make more sense together than huge jumble of tropes.

**Perspectives and Future Work**

In this study, we have defined tropes as well as their structures and have justified the existence of latent and cultural grammar rules for stories. What kind of outcomes can we hope from these results? What are the next challenges in this field?
A Look at Film Signature Extraction

This subsection is a short presentation of some results from our work on film signatures (Chou and Christie 2021) which was based on the results of this paper. The objective was to determine extractable features that would constitute the gist, namely the signature of a movie. This signature was then exploited for several applications.

In this regard, we started by picturing films as networks of tropes. For each movie, the nodes corresponded to the tropes, and the links were determined by the relationships between tropes in tvtropes ontology defined in this paper.

The analysis of these graphs of tropes led to promising results. Most central tropes were kept and constituted the signature. We proposed a new way to recommend movies by comparing signatures. When browsing the list of proposed recommendations, the results seemed conclusive. An example is presented Figure 14 for La La Land and Inception. The first movie is compared to other musicals and romances, and the second with other movies with complex stories or levels of reality.

Another example is the visualization of movies based on their tropes signature. The result shown in Figure 15 is informative since movies are clustered according to their genres and the similarity of their story.

Figure 13: An example of La La Land tropes network

Figure 14: The 10 most similar movies to La La Land and Inception among works with more than 100 tropes

Figure 15: A map of movies based on the similarity of their signature with t-SNE visualization. Action in blue, Comedy in pink

Tropes Prospects for Applications

More generally, the analysis of movie content is essential for many applications ranging from information retrieval to computational creativity. In this regard, tropes vocabulary is very promising as it would enable to fully break down movies or other works. For all such tasks, our previous study has shown that we will need to find the best-suited way of harnessing tropes heterogeneity. Here are two examples of prospects.

Recommendation Systems Film signature extraction briefly presented in the previous subsection is an innovative way to tackle recommendation systems task (Chou and Christie 2021). As of today, previous studies have used tropes as homogenous information to compare movies. Exploiting tropes informative and heterogeneous richness would enable to better understand users’ taste by detecting more detailed content similarities - whether they are about a plot twist, a personality trait, or a location for instance. Iron Man is a superhero action movie while Steve Jobs is a biographical drama. They share at first sight no common points, but their tropes indicate that they both mainly focus on a technovisionary philanthropist character.

We believe that this level of description could help tackle some of the next challenges of recommendation systems defined by Shi et al (Shi et al. 2014). For instance, long-tail
items are items that have low popularity and are therefore harder to compare with collaborative filtering methods. Content analysis with tropes would naturally overcome this problem. Cross-domain recommendation is another challenge that aims at predicting a user’s tastes for any media. Our study focuses on movies, but most tropes are common to various domains as the numerous media categories of tvtropes.org indicate. With tropes, similarities between works from different domains would thus easily be assessed.

**Story Generation** Thompson mentions tropes abstraction as a powerful and useful concept since it allows to break down stories into smaller and smaller components (Thompson 2018) and Guarneri uses James Harris’ partial narrative classification (Guarneri et al. 2017). None of them make full use of these abstraction layers, which could be a good starting point to structure and build a story from the general plot to the characters introduction. Such methods are useful creative tools as they offer a rich story vocabulary. But composition rules are still lacking to form a proper grammar and achieve story generation. Again, the analysis of films construction from their signatures might be a lead in this task.

**Future challenges**

Our analysis of tropes obviously presents significant lacks that must be addressed.

**Analysis of Levels, Characters and Media** First of all, our work mainly focused on tropes describing a movie as a whole. But such an approach could be applied at other levels of analysis. Working at scene-level or even frame-level sounds promising but the dataset lacks details for the moment. Characters could also be examined since tvtropes database includes tropes for more than 30,000 live action movie characters.

Besides, this study has only dealt with storytelling in live-action movies, but such an analysis could be performed on other media involving storytelling (animation, games, television, theatre, literature…). Including other media works would provide more examples to learn from and data to leverage on. Overall, the scope of our work could be extended in many ways.

**Temporal Analysis** One of our main lacks concerns temporal data. Knowing when a trope appears in an artwork absolutely but also relatively to another would open exciting new perspectives of analysis. Likewise, the duration of a trope is another central temporal feature yet to define and examine.

**Lack of data** These challenges are primarily a matter of the amount of data available. Therefore, the first stake will be to encourage tvtropes community to keep supplying the website database. The second stake will be to allow the community to add temporal information about the appearance of tropes. To this end, we could conceive a cross-platform application facilitating information entry.

**Tropes and Originality**

Tropes are often accused of being demystifying and cynical conventions ruining the abstract concept of art as they are often used by internet users to criticize works or to point out their impersonality. Besides, they seem to restrain creativity by simplistically breaking it down. To that we argue that tropes breadth and the countless ways to play with them offer a tremendous creativity. Some tropes can seem incompatible until they are exploited in the same plot or mixed to become a new trope. The website provides a tool generating random tropes of multiple narrative categories from which the user is challenged to build a story. Tvtropes community states that it makes no sense trying to build a story without any trope. Either the created story is of no interest or the latent processes are immediately inventoried, becoming tropes.

Tropes vocabulary is fired up by our imagination. It keeps growing as our cultures evolve and expands the boundaries of creativity. Tropes do not impose rigid patterns and methods. They should rather be considered as tools helping creators to grasp the content of our imagination, play with it, and reinvent it.

**References**


Conceptualizing Human-Computer Intersubjectivity to Develop Computational Humor

Paper type: Position Paper

Henna Katariina Paakki
Computer Science Department
Aalto University
Espoo, Finland
henna.paakki@aalto.fi

Abstract

Humor – as well as language in general – is by nature social and tied to a context. To better engage with context, computational humor could draw inspiration from the concept of intersubjectivity: the sharing of perspectives. This paper focuses on discussing the possible advantages of utilizing the concept of intersubjectivity to contextualize computational humor. Intersubjectivity in humor generation system design is discussed as a possible means of evaluation of the creative product, as well as a potential approach to generating more impressive humoristic content. Firstly, evaluation of computational humor has been wanting for more effective and versatile methods. To this problem, an implementation of sharing perspectives between the system and its users offers a viable solution. Secondly, approaches to humor generation are contrasted with interactive dialogue systems, to analyze how they contextualize humor. The comparisons show that well defined interactive design and evaluation methods that enable perspective sharing between the producer and the press would greatly benefit humor-generating systems. The final section theorizes on the possible foundations for modeling intersubjectivity in computational humor.

Introduction

Humor and its generation have been quite widely investigated in the field of computational creativity (Wen et al. 2015; Tyler, Wilsdon, and Bodily 2020; Valitutti et al. 2016). Many studies have concentrated on how to generate textual humor, i.e. short humorous texts, puns or jokes (Tyler, Wilsdon, and Bodily 2020; Valitutti et al. 2016). Although some approaches have succeeded in humorizing their human audience, most systems have remained too one-dimensional. Many models take a producer-centric cognitive approach to humor, operating on humor as a textual phenomenon that stems from linguistic or textual material via reproduction or recombination of language items like single words (Valitutti et al. 2016; Tyler, Wilsdon, and Bodily 2020). However, humor, like language itself that is used to convey the jokes, is not detached from our social life (Morreall 1983, p. 114-120). Following the Western socio-cultural and sociolinguistic view, it is in essence social, created in collaborative discourse with the surrounding social context (Bakhtin 1981; Vygotsky 1978). To create effective jokes that amuse our peers, we need to “know our audience”: to be aware of what knowledge is shared with the audience and what might be relevant to them. This allows meaningful communication with our interlocutors and places humor in an understandable context. A possibility for humor-generating systems to be able to contextualize humor can be found in the concept of intersubjectivity. Intersubjectivity refers to the relations between perspectives, and to a process of coordinating participants’ contributions to communication in joint activity (Gillespie and Cornish 2010). The central argument is that _humor creation cannot be separated from social context_; to generate contextualized jokes a system needs the concept of intersubjectivity: sharing perspectives with its audience and adapting its production to the feedback received.

This argument also brings us to the evaluation dimension of humor generation. According to Valitutti et al. (2016), there do not exist many effective evaluation methods for computational humor. From the creativity viewpoint of evaluation, systems that generate humorous texts can be creative if they portray novelty, value (Runco and Jaeger 2012), and surprise (Boden 1998; Bruner 1962). Surprise is considered here an especially important element for humor creativity, one that stems from the vantage point of producer and press perspectives (Jordanous 2016). As for the value dimension, humor is generally valued by human beings, for its intrinsic value or as a means of achieving social acceptance from others – in the case of computational generation mostly for its amusement-inducing effects. However, the question remains whether all automatically generated humor is equal in value – whether its intrinsic potential or its external evaluation by its audience might determine whether it indeed has value or not. As many aspects of humor are fitted according to the context in interaction with the expected audience (Carrell 1997a), concentrating on the latter evaluation dynamic would arguably be effective in developing computational humor. For this end, we may use the press perspective (Jordanous 2016), e.g. direct audience feedback like linguistic and humor cues, to improve computational humor.

Considering the points presented above, the evaluation dimension is a key issue in humor generation and cannot be evaded; evaluation is an essential part of both intersubjectivity as well as humor, because we (re)formulate humor content based on cues received from the press (Sacks...
1974). Thus, more interactive approaches to humor generation would arguably offer an effective means of evaluation for humor-generating systems. Moreover, humor is essentially dialogic in nature and thus should not be construed as a static standalone product. On the basis of these arguments, this paper will theoretically consider the possibilities and advantages to incorporating the intersubjective aspects of humor into its computational generation. Such an approach could allow better implementation of shared context in computational humor and thus lead to more impressive results.

This paper will concentrate on the following points:

- How interactive humor-generating systems could offer a means of solving some of the problems of computational humor evaluation.
- How humor-generating systems could benefit from the concept of intersubjectivity to generate more impressive products.

First, humor theory and earlier approaches to computational humor will be discussed. Second, intersubjectivity theory is introduced as a theoretical frame, and as a concept that guides the subsequent discussion on humor evaluation. Next, context representations found in earlier approaches to computational textual humor will be analyzed. These will be compared to examples of perspective sharing between humans and dialogue systems, to illustrate possible intersubjectivity in human-computer humor creation. This analysis serves to delineate the possible dimensions of intersubjectivity that could allow humor contextualization for computational models. The final section will propose a theoretical model of intersubjectivity for computational humor.

Earlier Approaches to Humor Generation

Theorizing Humor According to central theories of humor, it is constituted of three aspects: superiority, relief and incongruity (Morreall 2020). Superiority Theory states that we laugh at the misfortunes of others, reflecting our own feeling to be better. Relief Theory, on the other hand, explains that laughter is caused by the release of nervous tension (Morreall 2020). This may relate to taboo subjects embarrassing the reader and then relieving the tension. Incongruity Theory claims that a humorous effect is “achieved by the induction of incongruity in a playful context” (Valitutti et al. 2016). Incongruity, according to several definitions, involves a perception of incoherence, some contrast in meanings, or inappropriateness (Valitutti et al. 2016).

Humor research contributes to the building of a knowledge domain for a humor-generative system: this defines what constitutes its sense of humor in that it is used to delimit the outputs of the system. Thus it also guides the system’s humor perspective: knowledge of humor might for example be represented by incongruity, taboos, and contextual information within the system.

Approaches to Humor Generation Previous approaches to textual humor generation have often focused on the generation of short humorous texts. One such example is the system by Valitutti et al. (2016). As all humor-generating systems, it needs knowledge of language and the human concept of humor to be able to traverse the space of possible humoristic products. Their system is based on word replacements in SMS texts, and they try out various combinations of constraints that define how to implement a substitution. Firstly, the form constraint requires the substitute word to be (orthographically/phonetically) similar to the replaced word, e.g. “Which fart of town would you be in?” (as opposed to part). The taboo constraint requires the replacement word to be taboo-inducing. The context constraints restrict the to-be-replaced words to ones in text-final or second-to-last position, or make the substitution consistent with neighboring words. The authors have also evaluated the products after generation, by conducting a large-scale survey of their funniness in the eyes of a Crowdsource audience.

Another example of an approach to generating textual humor is Hahacronym by Stock (2003). Hahacronym takes an acronym and creates incongruity by replacing the original words forming the acronym with at least one new word that begins with the same letter as the original and comes from the same word type. However, it has some type of an positional connotation in relation to the original meaning, e.g. ACM: Association for Confusing Machinery (Stock and Strapparava 2003). Labuov and Lipson’s system (2012), on the other hand, creates two-liner jokes that provide both the set-up of the joke as well as the punchline, e.g. “Why is the computer in hospital? Because the computer has virus.” In quite a similar tone, Manurung et al.’s (2008) STANDUP system outputs punning riddles like “What do you call a cry that has pixels? A computer scream.” Computational creation of irony and sarcasm has also been a developing area of research, e.g. sarcastic bots that may have humorous implications (Veale 2018). However, I will concentrate on humor in a general sense, as considerations of irony and sarcasm are outside of the scope of this paper.

There are also systems that are somewhat more interactive, e.g. basing their production on human-rated examples (Winters, Nys, and De Schreye 2019). Also, Wen et al.’s approach (2015) moves closer to collaborative human-computer humor creation. Their system creates memes collaboratively with it users: users provide an evaluation by choosing the most amusing products (Wen et al. 2015). Overall, however, most approaches remain quite single-faceted as they do not retain much shared context between the producer and its audience. Producer-centric approaches have been criticized generally in the field of computational creativity for ignoring the important roles of the press as well as the creative process (Jordanous 2016).

Intersubjectivity in Humor

Language – as well as humor conveyed by the means of language – does not exist in a vacuum. Language is by nature social: in language use, meaning is shared in a discourse within a language community (Vygotsky 1978; Bakhtin 1986). In Bakhtinian terms, the discourse of the surrounding community is reflected in our own speech (Bakhtin 1986, p. 96), and following Bakhtin’s ideas as well as Vygotskyan philosophy, this dialogism is essential in learning the practices of our community and in learning to communicate
meaningfully (Vygotsky 1978). Humor is also a way of continuing community discourse (Carrell 1997a): it draws from what we have learned from the practices of the surrounding social environment, and from the language resources that allow us to communicate meaningfully.

Humor in interaction is not only play at linguistic form or recombination of linguistic items, but often a means to accomplish something by or within that interaction (Fine 1984). Humor is in essence social (Morreall 1983, p. 114-120). For instance, it might be used to intentionally misunderstand our interlocutor’s claims to make conversation less serious (Schegloff 1987). Using Jordanous’ terminology, the humorous product is created in a social process between the producer and the press (Jordanous 2016). A shared perspective is utilized to create something new.

The way humor is framed with ‘contextualization cues’ to signal the intended meaning (e.g. (para)linguistic features, intonation and tempo) also underlines its interactive functions (Gumperz 1982). The manner of thus testing our interlocutor’s perception of humor cues illustrates how we interactively assess and formulate humor with our interlocutors. Audience acceptance of humor cues can be seen as a "go-ahead" that allows the creation of humor (Sacks 1974).

It needs to be noted that humor is an extremely difficult art form. Perfectly understanding the complex social and interactive mechanisms related to successful creation of humor may not be possible in the near future, if ever. However, some relevant aspects of these dynamics are possible to grasp, for instance the type of knowledge that is needed to understand our social context, our audience and what type of humor might be fitting there.

The next subsection will discuss intersubjectivity theory as a framework for this paper. The following one will propose how intersubjectivity together with computational creativity evaluation models could enable new methods for humor evaluation.

**Intersubjectivity Theory**

Intersubjectivity theory offers viable concepts for establishing a social context in computational humor, and furthermore, a possible approach to co-creativity. It has been of great interest in psychology, philosophy, sociology, and linguistics, despite being a highly abstract concept; however, an operationalization of the domains of intersubjectivity to study both human and non-human intersubjectivity has been developed by Stevanovic and Koski (2018). Intersubjectivity is broadly understood as the sharing of minds (Stevanovic and Koski 2018), but often also as the sharing states, or perspectives (Gillespie and Cornish 2010).

Intersubjectivity has an emotional, deontic and epistemic domain (Stevanovic and Koski 2018). The emotional domain involves the sharing of emotional states: it entails both expectations of the interlocutor’s emotions or expressions of affect, as well as the ability to represent their emotional state (Stevanovic and Koski 2018, p. 47–49). Examples of emotional intersubjectivity include a mutual smile, or joint affective attitudes to objects. The deontic domain refers to organization and power in social interaction: what the norms of interaction allow us to expect from others (Stevanovic and Koski 2018, p. 49–52). For example, intersubjectivity can be found in the norms of social actions: pointing at an object invites joint attention to it, and asking a question invites a response. Finally, the epistemic domain consists of relations between subjects’ knowledge states. Assumptions of possible shared knowledge influences how we engage in interaction with them and how we interpret their actions (Stevanovic and Koski 2018, p. 52–54). For instance, epistemic intersubjectivity can be achieved in following the interlocutor’s gaze, noticing the target of their attention and representing what they might know about it.

To share perspectives with others people need to maintain a grasp on mutual knowledge and understanding (Clark and Brennan 1991). The shared knowledge assumptions (Clark and Brennan 1991; Stevanovic and Koski 2018) also apply to humor: jokes assume some shared perspective, be it knowledge of social norms like taboos, recent societal developments, or what one’s peer group or audience is like. Ideas are influenced by feedback from the interlocutor (Sacks 1974), which is why humor evaluation is an essential part of humor intersubjectivity.

Intersubjectivity is rooted in Theory of Mind skills (Stevanovic and Koski 2018). The Theory of Mind is often used to refer to an individual’s ability to infer another subject’s mental states that lead to actions taken (intentions, emotions etc.) (Baron-Cohen 2000). The theory has been used to study humor processing and appreciation (Samson 2012), but it has also been found to be important for the producer during a creative process (Magerko et al. 2009). However, Theory of Mind is only one aspect of intersubjectivity (Gärdenfors 2008). Although Theory of Mind capabilities are needed in humor intersubjectivity, here the focus is on the relations of perspectives: the perspectives of the producer and press. In the relationship between these two lies the possibility for surprising the press.

As other beings besides humans also possess capabilities for intersubjectivity (Stevanovic and Koski 2018; Bard 2012), some form of intersubjectivity seems plausible for computational systems. Intersubjectivity in human-computer interaction supposes some type of subjectivity to be possessed by the computational system. It could be argued that human-computer intersubjectivity cannot be considered due to a computer not having a mind, per se. However, it can also be regarded as a subject, acting of its own accord in relation to humans. The fundamental constraints a humor-generating system operates on can also be seen as directing its (humor) perspective that can be shared with an interlocutor. In the sense that intersubjectivity refers to sharing perspectives and building relations between them, intersubjectivity is possible between a computational system and a human: both parties can share perspectives on e.g. what objects are like, what they know of them, and associate some affect with them. Some examples of intersubjective coordination between human and computer can be seen in the field of chatbot and conversational agent development (Adiwandra et al. 2020).

In computational creativity and HCI, intersubjectivity has been employed as a concept for developing co-creation (Eicher et al. 2017) and human-AI interaction (Wang et al.
Intersubjectivity as a Means of Evaluation

In a Bodenian sense, computational textual humor creation might be novel in a combinatorial, exploratory or transformational manner (Boden 2004). Textual humor generation systems are often combinatorial (Valitutti et al. 2016; Stock and Strapparava 2003). However, having new combinations does not necessarily result in funny jokes. Many humor generation systems (Valitutti et al. 2016; Stock and Strapparava 2003; Labutov and Lipson 2012), do not use audience responses to reformulate their products and to produce humorous utterances that are possible and reachable, valid, appreciated, and also of high quality. There is a fine line between humor and nonsense. A system that produces anything by trial-and-error without precise knowledge of what constitutes high quality humor is not likely to be considered funny – as seen in audience reactions in Valitutti et al. (2016). To reach high quality concepts, producer and press perspectives need to be adjusted to better conceptualize what high quality space might consist of in a given context.

Evaluation Based on "Funniness" and "Humorousness"

Valitutti et al. (2016) argue that there do not exist a lot of evaluation methods for computational humor. Research on computational humor generation has traditionally evaluated automatically generated humor via "funniness" or "humorousness". More precisely, in computational humor, we may thus differentiate between a subjective experience of humor vs. the objective aspect of humorousness; there is a difference between subjective experience of appreciating humor or psycho-physiologically reacting to humor – i.e. "funniness" (Ruch 2007) – and the objective aspects of content being humorous, e.g. a text’s potential to induce humor appreciation – "humorousness" (Carrell 1997b). Funniness can be considered a dimension of "success" in automatic generation of humor, i.e. how strong a reaction the product induces in the audience. Although some researchers have evaluated the products of their humor systems (Valitutti et al. 2016; Tyler, Wilson, and Bodily 2020), most have not reiterated back to producing better products based on the evaluations.

Considering the lack of evaluation criteria or methods in computational humor, a humor-generating system could get evaluatory advantage from being more interactive: using audience response data as an evaluation method in order to generate humor that is more likely to receive an amused response. Such evaluation as used e.g. in (Wen et al. 2015) comes closer to incorporating an evaluation system for funniness already within the system, as compared to many other approaches. Arguably, we could venture further employing this type of an approach. Of course, a subjective human evaluation is possible in this manner, but potentially biased: it relies more on what individuals find funny as opposed to system’s potential for "humorousness", or the creativity of the system. However, if the success of the system is evaluated on the basis of the reaction achieved in the audience ("funniness"), then a subjective audience evaluation of a produced utterance as funny or not will arguably result in a funnier humor-generating system viewed from the audience perspective.

Creativity Evaluation of Computational Humor

Considering humor-generation in light of computational creativity illustrates how and why intersubjective elements could improve humor-generating systems. On the one hand, definitions of creativity in the computational creativity field allow a framework of evaluation of the content produced by a humor-generating system by appreciating the product’s creativity. On the other hand, they explicate the relationship between humor appreciation and the need for intersubjective referencing for a humor-generative system.

The alternating co-creativity model (Kantosalo and Toivonen 2016) and the dual generate-and-test model of creativity (Liu 2000) incorporate an interactive dimension within the evaluation of a creative product – between the producer and the audience. Firstly, alternating co-creativity bases itself on the Wigginsian idea of creativity as a search (Wiggins 2006). It views partners as taking turns in the collaborative creation of new concepts, to satisfy the requirements of both (Kantosalo and Toivonen 2016). Both parties rarely agree on what is relevant and interesting, so to reach high quality concepts and thus more transformational creativity, they need to adjust their views (Kantosalo and Toivonen 2016).

Liu’s dual version model of creativity (2000) depicts generation as an iterative process between the producer, the domain of the product, and the audience. Once content is produced, it will be evaluated by humans and the producer will turn back to production using the feedback and new knowledge to create better products. Considering the possibility of interactive humor-generating systems, the concept of audience is highly important: not only being aware of press perspective – shared knowledge or the background of the audience – but also considering the received feedback. A system can improve by getting a direct reaction (indication of mirth or laughter) from its audience, thereby judging the product a success or a failure.

To gain insights into how perspectives could be shared between a computational system and the press, the next section will look at how contextual information is represented in earlier approaches to computational humor. This is contrasted with examples of humor contextualization in chatbot–human interaction. The following subsection will then theorize on the possible foundations needed for intersubjec-
Intersubjectivity for Developing Computational Humor Co-creativity

Contextual void  While standalone jokes can be created without the existence of an immediate listener, they are textual products that have an expected audience and a frame of reference. They do not exist in a humorous vacuum, but instead have their own historical-contextual and intertextual references to socio-cultural phenomena that ties them to our social world in a meaningful way. However, many humor-generating systems suffer from a lack of social context.

Consider for instance some products of the system by (Valitutti et al. 2016), with the replacement first and the original replaced word in brackets:

Meena: You were trying to steer it elsewhere, I can see it.  
Human: do horses go to Harvard?  
Meena: Bovine sciences.  
Human: what did the cow study?  
Meena: I heard they go to college  
Human: cows go to college?  
Meena: I heard that a cow went to Harvard.

Despite the lack of a onomatopoeic tag, the automated joke would not be understood as a joke by anyone but the original user, because the texts have almost no context: they show no significant reference to the social context of the supposed reader. The products of Hahacronym (Stock and Strapparava 2003) suffer from a somewhat similar detachment problem. For instance FBI – Federal Bureau of Investigation – reformulated as Fantastic Bureau of Intimidation is not very funny if evaluated by "funniness", as very little context is given. Some perspective is shared in terms of what is assumed the audience might expect to come before "birthday" or what they are assumed to about the acronym FBI, but the perspective is not contextualized and thereby remains weak. To reach high quality humor concepts, humor needs to be placed within an understandable context.

Automatically generated two-liner jokes like (Labutov and Lipson 2012; Manurung et al. 2008) are a specific example of standalone texts that are are able to place the joke within a frame of reference. The set-up of the joke provides a context, and the punchline provides the element of surprise, creating thus the humorous effect. For example, in "Why is the computer in hospital? Because the computer has virus." (Labutov and Lipson 2012) the set-up provides the context by referring to a place and state we all are familiar with. The punchline takes advantage of the shared knowledge people have related to hospitals and computers. Continuing the incongruity between the concept of being in hospital and the common functions of computers, it achieves a playful effect. Thus, even short texts may be placed within a meaningful context. However, two-liners are a restricted form of humor. So, were we to move further to produce other types of humor it would make sense to consider the advantages of intersubjectivity in computational humor.

Sharing Perspectives Between Computers and Humans  Comedians who are skilled at coming up with good jokes in an ongoing dialogue with others, jokes that really amuse their interlocutor, have a knack for analyzing and representing social context (Quirk 2015). Thereby they are able to make intertextual references to the larger prevalent context as well as what was said previously in the conversation. In other words, comedy requires the skill to refer to a prevalent state of society or events that the audience can be known to be aware of, which relates to epistemic intersubjectivity (Stevanovic and Koski 2018). Great stand-up comedians, for instance, need to know their audience (Quirk 2015); they also possess excellent intersubjective ability, in reading the reactions of the audience and adapting their show based on cues from the press side.

For computers to know their audience, some level of intersubjectivity is needed, e.g. assumptions of share knowledge. Some examples of this can be found in implementations of chatbots or dialogue agents that are designed to interact with humans. A case in point is this multiturn dialogue between a human and the chat bot Meena (Adiwardana et al. 2020):

Human: I like how cute cows are  
Meena: Ah, interesting. You like how they are cute, right?  
Human: yeah, and they’re really smart  
Meena: I heard they go to college  
Human: cows go to college?  
Meena: I heard that a cow went to Harvard.  
Human: what did the cow study?  
Meena: Bovine sciences.

Although Meena’s main job is not humor-creation, and though this sort of lead in collaborative humor-creation may, as of yet, not be intentionally reproducible using the system, as an interactive system it is impressive. The extract shows acknowledgement of both the micro-context and relevance of a specific topic to the interlocutor, as well as knowledge of more general aspects of the surrounding society. Meena continues to discuss something the interlocutor indicates as an interesting or important topic; and also introduces world knowledge of a context-relevant place of study (context being the U.S.) that can be assumed as known to both. This shared knowledge also forms the building blocks of incongruity in the set-up of the joke: it can be assumed mutually known that cows do not attend college. However, the interlocutor gets the gist and collaborates to further the joke by assuming about the cow’s subject of study and also about the possibility of other animals taking up an academic career. The joke, in essence, is collaboratively created: Meena follows up on the subject of horses by exploiting the common association between horses and hay, and combining this with the phonetic/orthographic similarity between Harvard/Hayvard. As a new combination, “Hayvard” involves incongruity, but also portrays impressive intersubjective construction of a joke, as horses are introduced to the discussion by Meena’s interlocutor and not Meena themselves. The consideration of both audience expectations, world knowledge as well as the history of the discussion itself is what makes it seem like Meena "led the other person into the joke". They might have expected a perspective like...
"Yes, horses go as well, and they study Equitation science.", but the formulation of "Hayvard" is surprising here. It shares a new surprising perspective on what could next be associated with horses and Harvard. The context-awareness in the example is impressive, and illustrates a way of sharing perspectives between human and bot.

The ability to refer back in the conversation thus arguably contributes to the perception of humor creativity. Taking what has already been said, and repurposing it to create a novel concept that surprises the human user could well fit into the idea of alternating co-creativity of high-quality concepts. Although Meena’s Hayward joke as a two-liner would possibly be only mildly funny, arguably its effect is greater when tied to the context, surprising the interlocutor. It might be fruitful for a humor generation system to modify its views of humor e.g. based on topical cues or cues of surprise in the interlocutor’s reactions, to reach high quality humor concepts.

Some approaches to humor have already chosen to generate humor via chat bots (Augello et al. 2008; Sjobergh and Araki 2009). The former can both generate humorous sentences and recognize humoristic expressions. It uses a pre-existing open source chatbot Alice1. The recognition of humoristic language features is achieved by connecting the chatbot knowledge base to external resources (WordNet2 and the CMU pronunciation dictionary3). Although the jokes the bot tells are still two- or one-liner jokes prompted by user-selected topics, the approach is more interactive than many other models. The latter is based on retrieving jokes from joke database modules, and is thus not producing novel humor by itself. Computational memes have also been approached using a co-creative model: the system and a human user create collaborative humor using memes. During development, users chose the products that amused them the most. The choices were used to give more weight to successful products and to improve the system (Wen et al. 2015). However, this is but one approach to co-creativity in humor.

The question remains, can we do more: could we learn even more from the press to develop the system’s sense of humor? Could humor-generating systems come up with new creative jokes via sharing perspectives with humans?

### Defining intersubjective computational humor

Humor generating systems would greatly benefit from richer socio-contextual knowledge. Instead of only relying on human-ratings, it makes sense to improve the system’s understanding of humor and its context. For humor contextualization, a system needs the concept of intersubjectivity. The producer requires some shared perspective with the press, one that can be transformed to create a humorous effect. Next, I will discuss the operationalization of intersubjectivity, then considering knowledge and skills that a computational humor system would need, and providing a theoretical foundation for computational humor intersubjectivity.

---

1https://alicebot.org/
2MultiWordNet: http://multiwordnet.itc.it
3CMU Dictionary: http://www.speech.cs.cmu.edu/cgi-bin/cmudict

The formalization of intersubjectivity is not a trivial task. In fact, strict formalization of e.g. common knowledge has been deemed almost impossible by some scholars (Monderer and Samet 1989). However, seeing intersubjectivity as the sharing of perspectives – emotional, deontic and epistemic – it is possible to reach a practical approximate formalization of the concept. For instance, it is possible to approach common knowledge in a less restrictive sense as approximate common knowledge, or as common beliefs (Monderer and Samet 1989). In practical formalization of intersubjectivity, perspectives related to emotions, social norms and common beliefs can be assumed to be nearly mutual between the producer and the audience. This allows the sharing of a humor-contextualizing perspective and its transformation into a new surprising shared perspective.

The domains of intersubjectivity (Stevanovic and Koski 2018) can be seen to somewhat align with Relief, Superiority and Incongruity Theory. All the strategies used to create humor require some understanding of the audience perspective: e.g. to relieve audience tension or concern, we need knowledge about their emotional perspective; for superiority humor, we need knowledge of power relations within our social context, and which groups the audience belongs to; to induce incongruity, we need knowledge of what objects are like and what is expected of them. Of course, for instance incongruity in inappropriateness can be related to norms of social actions (deontic domain), so the matching aligns only partly. However, this illustrates the need of context and audience knowledge in creating humor using any of these strategies. Such knowledge allows perspective sharing: it is the implication of a perspective that is often used to contextualize a joke. This entails representing the associations people usually make based on the initial perspective, e.g. the word school often makes people think of some aspect of learning. Shared perspectives can be used to surprise the audience: humor is created at the vantage point of the perspectives of the producer and the press.

For a computational system this could mean inferring some emotion, social rule or (contextual) knowledge that can be shared with the audience, using it to contextualize humor, and then shifting the mutual perspective to surprise the audience by using relief, superiority or incongruity. Based on theory and analysis, the following elements are important and to some extent possible to operationalize: understanding of shared perspectives, sense of relevance, ability to refer back to contextual elements, cue detection (affective, epistemic, deontic), and awareness of social norms. The first two represent adaptable knowledge that is needed for humor contextualization; the latter three represent skills the system needs to be able to adapt to the context. Table 1 conceptualizes these possible foundations for intersubjectivity in human-computer humor generation. The list is not all comprehensive; it provides a theoretical consideration of features some of which could be used to improve computational humor generation.

First, a computational humor producer needs an understanding of possible shared perspectives: e.g. "world knowledge", knowledge about the interlocutor, and situational awareness. Humor systems could benefit from Web or news
mining or semantic resources like Web Ontology Language (McGuinness and Van Harmelen 2004) or WordNet, somewhat similarly to AI systems (Davis and Marcus 2015), thus accumulating domain knowledge. Such resources have already been utilized in digital storytelling (Peinado and Gervás 2006), as well as in AI creation of paintings (Colton 2012). Inference of shared knowledge could be based on the interlocutor’s utterances and characteristics, or world knowledge generalized within a limited social context.

Awareness of shared perspectives allows the capability to infer what information is relevant to one’s audience, and where mutual attention could be directed. In this respect, relevance is closely tied to intersubjectivity. To draw from Relevance theory (Sperber and Wilson 1986), the cognitive principle of relevance dictates that people pay attention to content that is relevant to them. To maintain audience attention the producer needs to be mindful of some degree of relevance. For instance, this could entail representing what concerns the interlocutor has on their mind and how to relieve the tension, or, as Meena did, discussing a topic their interlocutor showed interest in. Contextual knowledge regarding societal events and their relevance could be drawn from already existing knowledge or in interaction with the interlocutor by reading cues.

For computational intersubjectivity, it would be useful to maintain a recollection of past events within the larger social context and the ongoing interaction, as well as what the interlocutor is like. Simulating "memory" of past utterances, for instance, is possible using contextual machine learning models that base the next-utterance-selection on previous utterances. For example, like Meena, which is trained on context-utterance pairs where context is the previous utterance in a conversation (Adiwardana et al. 2020). The system’s knowledge-base could, for instance, be updated with knowledge about the interlocutor and their possible beliefs based on interaction.

Reading cues in interaction is essential for making inferences of shared perspectives: if the system is able to detect humor and surprise cues on the press side, as well as emotional, deontic and epistemic cues, it is possible to make informed inferences of perspectives that could be shared. E.g. the detection of humor cues (Farnia 2019) – or openings – and building on them would allow intersubjectivity between the system and the human, and thus a novel form of humor co-creativity. Certain linguistic and sentiment cues can also reveal how the interlocutor perceives the system (Wang et al. 2021). Understanding humor appreciation cues, on the other hand, would allow immediate humor-evaluation that could be used to improve the system.

Finally, an intersubjective humor generating system may also benefit from some awareness of the social norms of human interaction, e.g. how to respond to certain utterances (Enfield, Stivers, and Levinson 2010). This is not only related to respecting conversational rules, but also offers a possibility to humor the interlocutor, as permissible rule breaking of conversation maxims can be effective in introducing incongruity into humor (Lakoff 1982).

In conclusion, to be able to create high quality products considering the audience, the producer needs to accumulate knowledge based on what is constantly learned in interaction with the press. Humor creativity and intersubjectivity are essentially interwoven as illustrated in Figure 1. While the producer (a computational system) formulates a reachable and high quality product based on its knowledge at a given time, this is only the first step. Drawing more knowledge from cues (emotional, deontic, epistemic and humor cues) in interaction with the press, it can assess whether the product is indeed of high quality. It may also (re)assess what perspectives could be shared, accumulating more contextual knowledge and making inferences of shared perspectives and relevance. Based on the contextualization, the generative system reformulates what are now construed as possible

Table 1: Dimensions of intersubjectivity in computational humor.

<table>
<thead>
<tr>
<th>Function</th>
<th>Dimension of intersubjectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contextualization of humor</td>
<td><strong>Shared knowledge.</strong> Understanding of emotional, epistemic and deontic perspectives that are (approximately) shared with the audience.</td>
</tr>
<tr>
<td></td>
<td><strong>Sense of relevance.</strong> Inference of perspective relevance.</td>
</tr>
<tr>
<td>Inference of shared perspective and relevance</td>
<td><strong>&quot;Memory&quot;.</strong> Maintaining audience attention: to be able to refer to past events or content in the conversation or in the social context. To be able to learn from feedback, and about the audience.</td>
</tr>
<tr>
<td></td>
<td><strong>Cue detection.</strong> Detecting interlocutors’ attempts at and openings for humor, affective, deontic and epistemic cues, and feedback cues.</td>
</tr>
<tr>
<td></td>
<td><strong>Awareness of norms.</strong> Understanding social norms to interpret cues, and to follow social rules.</td>
</tr>
</tbody>
</table>

Figure 1: Intersubjective model of computational humor.
shared perspectives and thus high quality humor concepts.

Ideas for Future Paths

Humor is a social phenomenon, and so its computational creation should also be – by utilizing the intersubjective aspects of humor in its creation. Although some steps have been taken in this direction, especially the awareness of (larger) social contexts is still lacking in computational humor systems. The product is an interplay between the producer and the press – press engagement with either the larger societal discourse, or the discourse of the more immediate social group (Bakhtin 1981; Vygotsky 1978). Thus, taking social context better into account could improve computational humor. As argued in the proposed model, to strive for more intersubjective construction of computational humor, we need to know on some level what perspectives could be shared with the audience. The domains of intersubjectivity (emotional, deontic and epistemic) align quite well with humor theories, illustrating the type of audiencenowledge that is needed for different types of humor.

There are several directions humor generation research could take. For example, humor-generating systems could benefit from online news mining, semantic webs, or conversational data sets (Henderson et al. 2019; Li et al. 2017), especially ones that incorporate a robust labeling system for humor within conversation. Some of these have already been used in some cognitive models of humor to accumulate knowledge of objects and how they could be referred to in a surprising way (Stock and Strapparava 2003), but they could be used also to better understand the audience. Also, sentiment analysis or dialogue datasets that incorporate sentiment annotations of mirth or surprise could help understand interlocutor reactions (Li et al. 2017). Such resources could allow a system to learn about, for example, polysemy or adversarial humor (Veale, Feyaerts, and Bröne 2006), based on peer reactions in interaction.

On the other hand, research on humor or irony markers could be used to improve a generative system’s knowledge and perception of humor cues and what type of humor can elicit amused reactions from the audience (Farinha 2019). This could allow the system to also detect possible humor markers used by the interlocutor or potential openings for humor within a discourse. There already exist humor and irony detection systems and humor-identifying chat bots (Reyes, Rosso, and Buscaldi 2012; Joshi, Bhattacharyya, and Carman 2017; Augello et al. 2008), so it seems highly possible to create more interactive humor-detecting and generating systems. This type of decoding/encoding approach to humor-generation would allow for the system to detect humor cues or “openings” in the interlocutor’s contributions and to pursue co-creative humor-generation via perspective sharing instead of using ready jokes retrieved from a database. Detection systems could work to widen a system’s knowledge-base on what is considered as funny by humans, and allow inference of possible shared perspectives. There are also methods for detecting emotional cues, dialogue acts or intent, and stances to epistemic content, for instance (Stolcke et al. 2000; Liu 2012; Ferreira and Vlachos 2016); methods, which could contribute to computational intersubjectivity.

Finally, interactive chat bots provide examples of contextualization that could be learned from. Being able to share perspectives and learn from them has a lot of potential for computational humor, e.g. because the system can get direct press evaluations. This could lead to new modes of evaluation that emphasize audience reactions even more than earlier models, like interactively bidding possible funny products to the user and then returning to production based on user reactions. User evaluations need not be limited to ratings or training on user selected jokes, but could also involve interpreting cues of user reactions like amusement or mirth.

Limitations and Ethical considerations The theoretical position taken here does not mean, however, that the non-interactive approaches dealing with short texts do not have their advantages. These offer valuable insight into how the system’s humor constraints could be designed. They could also be embedded into a more comprehensive system, which could provide them with more context and allow versatile humor techniques via using specific joke-creation modules. It is also noteworthy that the approach suggested in this paper may have its challenges, as some implementations of it may be too all-encompassing if not delimited in a sensible manner. Experimentation with more constricted generation of short humorous texts can be, as an initial approach, more clearly restricted and manageable. However, I argue that implementing some aspects of the model of intersubjectivity could develop co-creativity in computational humor.

Perhaps at the moment the more general concept of co-creativity might allow an easier approach to developing computational humor between humans and computers. However, there is something interesting that could be learned from the concept of perspective sharing, specifically for computational humor. Namely, how co-creativity in humor could be implemented: not only through using audience-rated examples in training the system, but by developing the system’s understanding of contextual humor. Humor is created at the vantage point of the perspectives of the producer and the press. Thus, understanding the audience is important for developing systems that better interact with the press. Although humor is a very difficult art form, such knowledge improves the system’s chances of creating humor that is appreciated.

Humor has an interactive social function: it can help people reach common ground, or divide them, especially when considering superiority-type humor that feeds off a certain subgroup that is made into the butt of a joke. Furthermore, since language can work as a reflector of ideology, humor can as well. Thus, humor systems should be built keeping the value aspects of the surrounding society in mind; as a means of communication, humorous products might show engraved ideological positions and might thus have an effect on the receiving end, e.g. biases related to gender, racism, hate speech, offensive language, and revealing of sensitive information. An unfortunate case in point is the Microsoft Tay chat bot. Thus, careful ethical considerations are needed in interactive system design.
Concluding remarks  In conclusion, computational humor should not consider textual instances of humor as context-detached series of words or reproduced combinations of words, which might be considered funny by humans — or not. The opportunistic manner of producing reachable concepts that may be valid in a technical sense does not in many cases result in high quality concepts. A more impressive co-creative system will enable a sharing of perspectives between itself and humans, by considering its audience and the context they are both placed in. Collaboratively re-formulating approaches to humor based on the contextualization of what is shared with the audience, such a system could achieve a more transformational level of creativity as compared to many systems showing combinatorial creativity. The importance of the concept of intersubjectivity in humor-generation is to allow more fine-grained creativity in computational humor: to approach humor as a social construct, which takes its most impressive forms in the sharing and transformation of perspectives in surprising ways.

References
Bakhtin, M. M. 1986. Speech genres & other late essays. Austin, TX: University of Texas.
Inspiration through Observation: Demonstrating the Influence of Automatically Generated Text on Creative Writing

Melissa Roemmele
Language Weaver (RWS Group)
Los Angeles, CA, USA
mroemmele@sd.com

Abstract

Getting machines to generate text perceived as creative is a long-pursued goal. A growing body of research directs this goal towards augmenting the creative writing abilities of human authors. In this paper, we pursue this objective by analyzing how observing examples of automatically generated text influences writing. In particular, we examine a task referred to as sentence infilling, which involves transforming a list of words into a complete sentence. We emphasize “storiability” as a desirable feature of the resulting sentences, where “storable” sentences are those that suggest a story a reader would be curious to hear about. Both humans and an automated system (based on a neural language model) performed this sentence infilling task. In one setting, people wrote sentences on their own; in a different setting, people observed the sentences produced by the model while writing their own sentences. Readers then assigned storiability preferences to the resulting sentences in a subsequent evaluation. We find that human-authored sentences were judged as more storable when authors observed the generated examples, and that storiability increased as authors derived more semantic content from the examples. This result gives evidence of an “inspiration through observation” paradigm for human-computer collaborative writing, through which human writing can be enhanced by text generation models without directly copying their output.

Introduction

Creative text generation is a significant focal point at the intersection between computational creativity and natural language processing research. The goal behind much of this research is to understand and simulate human creative writing abilities. There is also increasing interest in using this work to augment human creativity. This objective has become especially visible given recent advancements in systems that can directly interface with human-authored text.

Many existing creative text generation systems can be applied to facilitate human authoring, even if they are not explicitly presented in this way. The clarity of this use case can largely depend on how the system is evaluated. There is no standard design for such evaluations of benefits to human authoring. Much work uses the convention of comparing generated output to human reference output for a given task, either by comparing the features of the text itself or comparing relative human judgments of it. Success by this standard is based on how well the system simulates human writing. One could theorize that the more a system writes like a human, the more it will be able to help other humans write, but further empirical exploration of this is needed. Alternatively, systems that explicitly aim to support human authoring are often evaluated in the context of interactive applications where authors can elicit generated text. Here, the quality of the model can be evaluated according to authors’ interaction with the generated output.

In this paper, we focus on an “inspiration through observation” paradigm for human interaction with generated text. In many application settings for text generation, this human interaction is dynamic, with system output changing frequently in direct response to user choices. While discovering the best interaction paradigm is a critical objective of research on authoring support, here we minimize the role of user control over the generated text in order to assess the impact of merely observing the text. Authors see examples of generated text that fulfill a particular authoring objective, and they repeat the same task on their own. We compare human authoring outcomes in the absence and presence of these generated examples. This broad methodology could be applied to probe the ability of any system for aiding authoring, even systems that have not previously been assessed for this use case.

Our exploration of this paradigm focuses on a particular authoring task, sentence infilling, and a particular authoring objective, which we term storiability. Sentence infilling involves expanding a list of words into a full sentence. In our version of this task, the sentences we elicit can be viewed as story excerpts. The construct of storiability is related to previously discussed ideas such ‘storiness’ by Bailey (1999), which pertains to the success of a story from a reader’s perspective. We define storiability as the degree to which an excerpt (here, a single sentence) alludes to an appealing story. Even though this is a broad definition, we operationalize it through specific instructions in our experiments. Through our experiments we find that observing automatically generated examples of our sentence infilling task helps people...
better fulfill the storiability authoring objective. This provides evidence for a general inspiration-through-observation framework by which generation systems can improve human authoring.

**Background**

As artificial intelligence has progressed, so has the development of Creativity Support Tools (CSTs). CSTs are digital applications intended to augment human abilities in creative endeavors like visual and performance art, music, and writing (see Frich et al. (2019) for a review of several applications). CSTs for writing in particular have been boosted by recent advances in natural language generation, making it possible for systems to interface with any unconstrained human-authored text. This includes figurative language like poetry (Kantosalo, 2019) and metaphors (Gero and Chilton, 2019). Advances in story generation (e.g. Fan, Lewis, and Dauphin, 2018; Martin, 2021) have been showcased by the increasing development of CSTs that support authoring in the narrative domain. One design pattern for these systems involves authors querying a generation model for a “suggestion” that can be integrated into their text (Clark et al., 2018; Khalifa, Barros, and Togelius, 2017; Manjavacas et al., 2017; Roemmele and Gordon, 2018b). This enables analysis of what users choose to do with the generated text (e.g. retaining or deleting it) and how their choices are affected by the features of the text (Akoury et al., 2020; Roemmele and Gordon, 2018a; Clark and Smith, 2021).

Human-computer interaction studies have compared people’s writing with and without the use of AI-based tools, showing that these tools do change how people write. Exist- ing work has examined the effect of word and phrase predictions for content like image captions (Arnold, Chauncey, and Gajos, 2020), emails (Buschek, Zürn, and Eiband, 2021), and movie reviews (Bhat, Agashe, and Joshi, 2021). For more open-ended creative writing tasks, most research has focused on optimizing and assessing how much people favor the generated content. What is needed is more experimental comparison of how the use of CSTs changes the authoring outcome as perceived by readers. Mizrahi, Yardeni Seelig, and Shahaf (2020) recently pursued this for the specific task of creating neologisms (i.e. new words). In their work, people wrote neologisms before and after observing automatically generated examples. The results showed that observing these examples helped people produce better neologisms in terms of their perceived creativity. In this paper, we follow a similar approach to examine the intervening effect of generated examples for the sentence infilling task.

**Sentence Infilling**

We focus on the specific task of sentence infilling to evaluate our hypotheses about authoring. Given a sequence of input words (e.g. “he town rain”), which we refer to as a “prompt”, the infilling task expands the sequence into a complete sentence by inserting additional words (e.g. “he rode his bike to town in the pouring rain”). We created a dataset for this task and trained an automated model on it, as detailed below.

**Overview**

Text infilling, alternatively known as expansion or elaboration, has recently attracted significant attention for multiple types of corpora (Donahue, Lee, and Liang, 2020; Fedus, Goodfellow, and Dai, 2018; Huang et al., 2020; Shen et al., 2020). There are different configurations of this task based on the length of the text to be infilled. For stories, some work has focused on inserting sentence-length sequences that connect passages (Chandu, Dong, and Black, 2020; Ippolito et al., 2019; Mori et al., 2020). A more constrained version of infilling turns it into a cloze (i.e. fill-in-the-blank) task where infilled segments are single words or short phrases. Our infilling model outputs a single sentence given a sequence of words, but no assumptions are made about the number of words to infill. This design is reflected in existing work applied to creative authoring support (Özbal, Pighin, and Strapparava, 2013; Safovich and Azaria, 2020), but it has yet to be examined how automatically infilled sentences affect human performance of this task.

**Dataset**

We are not aware of any datasets that mirror the design of our particular infilling task, by which sentences can be generated from any arbitrary sequence of words. However, it is easy to simulate an infilling dataset using existing corpora. Given that the task is framed in the context of storytelling, we obtained 10,000 English-language stories from a variety of genres in the BookCorpus (Kobayashi, 2018). We segmented each story into sentences\(^2\), filtering sentences with less than ten words. To derive pairs of prompts and infilled sentences, we randomly dropped between 60-100% of words in each sentence. We required that the resulting ablated sentence consist of at least 50% content words (i.e. nouns, verbs, adjectives), since function words that convey little semantic meaning (i.e. pronouns, prepositions, determiners) are more frequent in text. The ablated sentences became the prompts used as the source inputs to the model, whereas the corresponding original sentences were the target infilled outputs generated by the model. The mean number of words in the prompts and infilled sentences was 4.86 and 19.19, respectively. These pairs were divided into 34,172,128 training instances, 897,473 validation instances, and 894,484 test instances fully held-out during training.

**Model Design**

Our infilling model\(^3\) is a Transformer language model (LM) (Vaswani et al., 2017), which is currently a popular architecture for many machine learning approaches to language generation. Figure 1 broadly illustrates the model. Our scheme

---

\(^2\)All linguistic processing steps used to derive this dataset, including sentence segmentation, word tokenization, and part-of-speech tagging, were performed with the spaCy library: spacy.io

\(^3\)We used the Texar-PyTorch library for implementation: texar-pytorch.readthedocs.io. Additional hyperparameter settings included: maximum epochs = 100, batch size = 32, gradient accumulation over 8 steps, validation every 25,000 steps, early stopping after 25 consecutive rounds of no validation improvement, static learning rate = 0.001, maximum gradient norm = 1.0.
for applying this architecture to infilling is closely related to that described in Donahue, Lee, and Liang (2020), with one main distinction. Their approach uses designated tokens (i.e., [BLANK]) in the input sequences to indicate the position where text should be infilled in the output. Alternatively, we only represent prompt words in the input, without any explicit signal for where text should be infilled between prompt words. As in the cited work, we initialize the model with weights from pretrained GPT-2 (Radford et al., 2019) as a means of embedding general knowledge of English text. GPT-2 has been highlighted for its potential to generate creative text (See et al., 2019; Dathathri et al., 2020). We used the “small” version of GPT-2 (117M parameters) and also the corresponding GPT-2 tokenizer to represent all text as subword tokens. We concatenated each prompt and corresponding infilled sentence together as a single token sequence, using designated tokens to signify the start ({}) and end ({}) of the prompt. To avoid memory errors, we set a limit on the size of the sequences by truncating prompts to the first 25 subword tokens and target sentences to the first 75 tokens. We then fine-tuned the pretrained model for the infilling task by training it on the dataset described above, using the maximum likelihood estimation loss function that is standard for training neural LMs. Our only variation from standard LM training was that we optimized using only the loss for the tokens in the target infilled sentences, and did not compute the loss of the source prompt tokens. This simulates an encoder-decoder scheme which decodes target text from the encoded source input; here the LM functions as both an encoder and decoder, which significantly reduces the number of parameters in the model. We monitored perplexity on the validation items in order to end training when perplexity stopped improving. In inference mode, the model progressively sample from the LM probability distribution and append the resulting token to the sentence, until the end-of-sequence token (i.e., [EOS]) is generated.

**Figure 1: General architecture of sentence infilling model**

**Authoring Experiment**

We next designed a human authoring task that integrates our trained infilling model. To broadly summarize this process detailed in this section: we selected certain prompts from the test partition of our dataset and generated infilled sentences for them. We then elicited human-authored infilled sentences for these same prompts. People produced sentences in two conditions. In the first, they simply wrote sentences for each prompt. In the second, they were shown the sentences generated by our model for the same prompts and wrote new sentences. We explain each of these steps below.

**Prompt Selection**

We selected prompts from the test set with exactly three words. This particular length value was picked based on intuition. Fewer words approximates unconstrained generation rather than infilling, while more words simulates a constrained fill-in-the-blank task. We excluded prompts derived from dialogue sentences (i.e., those containing quotation marks). Dialogue can pose issues for sentence segmentation (e.g., “he said.” may be segmented as a separate sentence from its adjacent quote). We also excluded prompts containing punctuation, numerical digits, named entities, or word tokens not recognized in the DistilBERT (described below) tokenizer vocabulary. Finally, we excluded prompts with more than one function word (e.g., pronouns, prepositions, determiners). By applying these constraints, we expected the prompts to give clear semantic cues for the infilled sentences. The resulting selection consisted of 23,005 prompts.

Since the process for deriving prompts involved random ablation of full sentences and the position of the ablated words varied, we theorized that even prompts of the same length require different degrees of infilling to yield grammatical sentences. For example, the prompt “his, body, relax” already resembles English syntax, and thus it would only take a single infilling word to produce a grammatical sentence (e.g., “His body could relax”). In contrast, it is possible but harder for native English speakers to find a single infilled word that could transform the prompt “peculiar, rob, more” into a grammatical sentence. Accordingly, we expected that the difficulty of the task would vary according to the degree of required infilling for a prompt. We designed an approach for automatically scoring this difficulty. For each selected prompt, we scored the probability of each of its word tokens according to the Masked LM configuration of DistilBERT (Sanh et al., 2019). A Masked LM is well-suited for this measure because it is specifically trained on a fill-in-the-blank task to predict the likelihood of words according to their context. We used the average of the prompt token probabilities to represent the inverse difficulty (i.e., easiness) of a prompt. We theorized that high-probability prompts are easier in infill since they are already probable sequences, whereas low-probability prompts require more infilling to become probable. We assigned the difficulty label “easy” to the 10% highest-probability prompts and the label “hard” to 10% lowest-probability prompts, yielding 2,301 prompts for each difficulty level.

**Generated Sentences for Prompts**

We then applied the trained model to produce infilled sentences for the selected prompts. We generated five infilled sentences per prompt, using the decoding method of nucleus (top-p) sampling with p = 0.7, based on the parameters recommended by DeLucia et al. (2020) for generating narrative text. The generated output followed constraints
consistent with the human authoring instructions described below. In particular, the generated sentences had to contain all prompt words in the same order as they appeared in the prompt. Prompt words were allowed to be capitalized in the sentence. Sentences had to consist of at least seven word tokens but no more than fifty. We additionally restricted sentences with quotation marks and missing end-of-sentence punctuation (i.e. by requiring the last character to be non-alphanumeric), since this may signify the sequence is not a complete sentence or combines multiple sentences (e.g. quoted dialogue). We filtered sentences with adjacently repeated words (this is a frequently observed issue with neural LMs). Finally, we promoted the diversity of the five sentence outputs for a given prompt by filtering any sentence with 60% or more of its words already appearing in previously generated sentences for that prompt. All of this criteria was satisfied by continually generating sentences for a prompt until it yielded five acceptable outputs. As a last step that we performed through manual review, we filtered any items where the prompt or generated sentence contained profanity or offensive content. This was done to minimize potential risk of harm to participants in the experiment. The final set consisted of 2,205 easy items and 2,189 hard items.

Human Authoring Task

We then conducted a human authoring task utilizing the selected prompts and generated sentences. Participants were instructed that they would be shown a list of three words (the prompt) and would write two unique sentences containing those words. They were presented with some manually written examples of infilled sentences. The instructions emphasized that they should “try to write sentences that evoke a story someone would be curious to hear”, which activates the construct of storiability that we focus on in this work. The authors’ sentences were required to obey the same prompt token order, length, and end-of-sentence punctuation constraints as the model output, which we enforced through the user interface. In the first stage of the task (the PRE stage), each author wrote two sentences for five prompts, which were randomly sampled from the “easy” and “hard” categories. In the second stage (the POST stage), authors were again shown the same five prompts and wrote an additional two unique sentences for each. This time, the five generated sentences were shown to them as examples they could reference while writing. Their sentences were required to be different from the examples. Figure 2 shows an example screenshot of the interface for this exercise.

The presence of the generated examples was the only variable that differed between the two stages. In both stages, after submitting the sentences for a single prompt, participants were shown generated text passages (described as “stories”) that each began with the sentences they wrote. These passages were generated by the original pretrained GPT-2 (not the infilling model). Passages had a maximum length of 75 words, and only the first complete sentences within this limit were displayed. The instructions informed authors that writing more interesting sentences would yield more interesting stories. However, this component of the task was not an experimental variable, since it was not varied between the two stages. This feedback was simply intended to incentivize authors to write more storable sentences.

6This was implemented as a ReactJS + Flask web application.

7Using the model interface provided by HuggingFace transformers; generated using nucleus sampling with p = 0.7
<table>
<thead>
<tr>
<th>Prompt</th>
<th>Difficulty</th>
<th>PRE Sentences</th>
<th>POST Sentences</th>
<th>GEN Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>walking and seeing</td>
<td>easy</td>
<td>1. The little children enjoyed walking through the zoo and seeing all the different animals. 2. The boy’s favorite activity was walking to the marina and seeing all of the boats in the water.</td>
<td>1. After being released from prison for a crime he didn’t commit, the old man was thoroughly enjoying walking through the city and seeing how the world had changed. 2. The woman cried when she saw her little girl walking and seeing for the first time after she got her new glasses.</td>
<td>1. She felt the urge to cry, but she kept walking and seeing no sign of it. 2. He was walking in front of the stove and he looked down on the ground seeing what was going on. 3. We were walking in and were immediately upon seeing what the neighbors had in store. 4. She was walking with a friend, and she just happened to be seeing a man, a man, and he was going to kill her. 5. She could hear men walking up and down the alley, and she didn’t know what they were doing, but she couldn’t deny seeing the resemblance.</td>
</tr>
<tr>
<td>nose pushed see</td>
<td>hard</td>
<td>1. The sled dogs nose was in the air as it pushed through the snow to see his owner. 2. I held my nose and pushed the stinky garbage can to the curb to see if I can catch the garbage man in time.</td>
<td>1. The dog, using his big nose, pushed the front door open to see if his owner was home. 2. The boy held his nose to stifle a sneeze but the involuntary reflex pushed his head forward, watering his eyes and making it hard for him to see.</td>
<td>1. The man’s nose was being pushed up and down, and as he moved closer to the screen, the image started to dawn on him, and he was shocked to see his father lying on the ground, dying. 2. He cleared his throat, the same way he had when he had slapped the back of his head and nose, then pushed himself away, but he was careful not to let her see his anger. 3. When he saw his own nose in the white sordid mess, he pushed off his seat to see it for himself. 4. He kissed her nose and pushed the sleeve of her shirt back to see what she was thinking. 5. A stray nose-bleed might be pushed up, but I couldn’t see anything out of place.</td>
</tr>
</tbody>
</table>

Table 1: Examples of authoring blocks. Each block consists of sentences written by a single author before (PRE) and after (POST) observing the generated (GEN) example sentences.

We recruited participants for this task through Amazon Mechanical Turk® (AMT), a crowdsourcing platform. 23 authors from majority native English-speaking countries were each paid $10 based on an estimated completion time of 45 minutes to 1 hour. The result was a dataset of authoring blocks, with each block consisting of a prompt shown to an author, their two sentences written before observing the generated examples (PRE), their two sentences written after the observing the generated examples (POST), and the five generated examples they saw (GEN). Examples of authoring blocks are shown in Table 1. With each author responding to five unique prompts, this yielded 115 blocks. We filtered six blocks where at least one sentence response (PRE or POST) was revealed to actually consist of multiple sentences (since this wasn’t straightforward to check through the interface during the task). This ultimately resulted in a set of 109 blocks to be used for evaluation, 53 for easy prompts and 56 for hard prompts.

Evaluation of Authoring Experiment

In line with the objective of the authoring task, we conducted a judgment task to evaluate readers’ perceived storiability of the sentences in the authoring blocks. This resembles story generation evaluations where people are asked which one of a set of stories they most prefer reading (e.g. Fan, Lewis, and Dauphin, 2018). For each of the 109 blocks, we gathered all unique combinations of the two PRE sentences, two POST sentences, and the first two of the observed GEN examples in that block, yielding 872 judgment groups (109 * 2 * 2 * 2 = 872). Thus, each judgment group consisted of a PRE, POST, and GEN sentence aligned to the same prompt and author. We designed a questionnaire targeting the relative storiability of the sentences in each group. Raters were instructed to “imagine that each sentence [in the judgment group] is an excerpt from a story and pick the one that makes you most want to read that story”. Only the sentence text itself was shown, and the sentences in each group were randomly ordered. We recruited 16 participants from majority native English-speaking countries through AMT to
felt meet again

Jenna felt a spooky sense of deja vu and felt that she was about to meet a familiar stranger yet again.

Bonnie felt a syrupy sentimentality and nostalgia and wanted to meet Raphael again.

I felt so relieved to meet you again.

regard sorts prevent

He had no regard for his own safety, a maverick of sorts, which did nothing to help prevent him from oft getting injured.

In regard to the message, there were all sorts of interpretations that could be made, so she asked for clarification to prevent misunderstandings.

A lower regard may come to any type of treatment that may result in a delay of sorts in order to prevent future evidence of therapy.

servants early life

It’s sad when people have servants that have to wake up early and do everything for someone else without having a life of their own.

They became servants at a very early age after having a difficult life and losing their parents.

But, yes, there were two excellent servants from a very early age in the village, who could carry the life of an even younger man.

hoping questions few

I was hoping I could find the answer to my homework questions, and after a few minutes I found them by doing a simple Google search.

She pored her thoughts, fears, and dreams into her diary, hoping that by writing them down, she could answer the vexing questions of life that few people ever really understood.

They were hoping to avoid answering any questions for a few days.

quickly and joined

There was a bird that quickly fell from the sky and joined with the ground.

The car quickly entered the lane and joined with the traffic.

The nurse quickly packed up the case and joined him.

arms awkwardly car

The arms hung awkwardly out the window of the car.

His arms flung awkwardly as the police slammed him up against the car to cuff him.

Sue wrapped her arms around his neck, pulled him awkwardly out of the car, and then pushed him down the long, steep driveway.

Table 2: Examples of judgment groups. The bolded sentence in each group was selected by both raters as the most storiable.

rate subsets of 55-56 judgment groups, with each paid $5 for an estimated completion time of 25-30 minutes. There were two raters for each subset, yielding a total of 1,744 responses ($48 for authoring blocks with easy prompts and $96 for hard). Examples of judgment groups are shown in Table 2. In these examples the bolded sentence was picked by both of its raters as the most storiable among the group.

For the results described below, we discuss judgments in terms of storiability preferences. In particular, each response is a single data point where the most storiable sentence selected by the rater was labeled as “Preferred” and the other sentences in the judgment group were labeled as “Not Preferred”. All data points have equal weight in the analyses.

Results

Human versus Generated Storiability Table 3 shows the normalized distribution of storiability preferences across the PRE, POST, and GEN sentences, along with their raw number of “Preferred” votes. Note that if preferences were randomly distributed across these three sets, each would approximate 0.33 (one-third) of the distribution. The numbers show that people notably preferred human-authored sentences (both PRE and POST) to GEN sentences (statistically significant at \( p < 0.05 \))

In contrast with human authoring, the infilling model did not receive any explicit instructions about the storiability authoring objective. The model was simply trained to generate sentences that appeared in stories. We can guess that the training sentences observed by the model are not all equally likely to be perceived as storable. It is possible that this is why raters favored human-authored sentences over the generated ones. However, even generated text designed to mimic human writing objectives often does not meet this standard (e.g. Lin et al., 2020), so the difference in preferences is not simple to interpret. The focus of this particular paper is not on comparing the relative quality of human and generated text, but on whether generated text can alter the quality of human writing. Thus, the rest of our analyses concentrate on this question.

Prompt Difficulty Table 4 shows the effect of difficulty on the number of infilling words people used to connect the prompt words. The human-authored sentences for the hard prompts had significantly more filled words between
Table 3: Distribution of storiability preferences

<table>
<thead>
<tr>
<th>Preferred PRE</th>
<th>Preferred POST</th>
<th>Preferred GEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.356 (621)</td>
<td>0.365 (636)</td>
<td>0.279 (487)</td>
</tr>
</tbody>
</table>

Table 3: Distribution of storiability preferences

Prompt words compared with easy prompts ($p < 0.05$). This validates the expected difference between these conditions, suggesting that hard prompts required more authoring effort.

<table>
<thead>
<tr>
<th>Difficulty</th>
<th>Infilled Words</th>
</tr>
</thead>
<tbody>
<tr>
<td>easy</td>
<td>3.035</td>
</tr>
<tr>
<td>hard</td>
<td>4.317</td>
</tr>
</tbody>
</table>

Table 4: Mean number of words between prompt words in human-authored sentences according to difficulty

**Prompt Difficulty and Storiability** Table 5 shows the distribution of preferences for PRE and POST sentences grouped by difficulty level. We found that POST sentences had higher storiability than the PRE sentences, but only for hard prompts ($p < 0.05$). Thus, people were more likely to write storiable sentences for these prompts after observing the GEN examples. The result for easy prompts showed a tendency towards the reverse pattern, but the difference in this case was not statistically significant. Based on this result, we focus our subsequent analyses on the items associated with hard prompts. We return to some discussion of this interaction effect regarding difficulty in the next section.

<table>
<thead>
<tr>
<th>Difficulty</th>
<th>Preferred PRE</th>
<th>Preferred POST</th>
</tr>
</thead>
<tbody>
<tr>
<td>easy</td>
<td>0.384</td>
<td>0.354</td>
</tr>
<tr>
<td>hard</td>
<td>0.329</td>
<td>0.375</td>
</tr>
</tbody>
</table>

Table 5: Distribution of storiability preferences for human-authored sentences by difficulty

**Influence of Generated Examples** The higher preference for the POST sentences suggests that observing the GEN examples had some impact on authors. One could consider other interpretations: for example, maybe authors were simply better at the task in the POST stage after a round of practice in the PRE stage. To investigate this, we first determined whether any influence of the GEN examples could be quantitatively detected in the POST sentences. There are many different features that could be used to quantify this influence. Here we focused on whether authors incorporated semantic content from the examples they observed. We assessed this using a quantitative measure of semantic similarity between sentences based on vector representations given by a pretrained language model. Intuitively, pretrained LMs are expected to produce similar vector representations for sentences with a similar meaning. This representation should transcend the lexical level, so that even sentences with few words in common can have a high similarity score if their respective words in context are synonymous. We computed semantic similarity between the PRE and GEN sentences, and then separately between the POST and GEN sentences. Since the GEN examples were not shown in the PRE condition and thus could have no influence on the PRE sentences, any significant difference in this measure between the PRE and POST sentences can be attributed to authors observing the GEN examples.

We computed the cosine vector similarity between sentences encoded with the DistilBERT\textsuperscript{10} LM. For a given prompt, the similarity score for a human-authored sentence $h$ is its maximum similarity over all GEN examples $g_s$ for that prompt, i.e. $\text{score}(h, g_s) = \max_{g \in g_s} \text{sim}(h, g)$. We select the maximum because there may be one GEN example in particular that most influences a given sentence.

Table 6 shows the mean of this similarity measure for the PRE and POST sentences. POST sentences had higher similarity to GEN sentences ($p < 0.05$), confirming that the GEN examples had semantic influence on the authors’ writing.

**Influence and Storiability** After verifying that the difference between the PRE and POST conditions can be attributed to semantic influence from the GEN examples, we examined whether this influence was related to the higher storiability of the POST sentences. Table 7 demonstrates that sentences preferred as more storiable were also more semantically influenced by the GEN examples, as indicated by the higher similarity scores for the Preferred sentences ($p < 0.05$). Thus, by incorporating some degree of content from the GEN examples, people tended to better fulfill the authoring objective. Table 8 gives some examples of judgment groups where semantic influence can be qualitatively observed in the POST sentence. The GEN example with the most influence is shown (i.e. the one most similar to the POST sentence), and we comment on the subjective evidence of their similarity. These results encourage future opportunities for explaining the exact mechanism underlying semantic influence. We discuss this further in the next section.

<table>
<thead>
<tr>
<th>Condition</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRE</td>
<td>0.921</td>
</tr>
<tr>
<td>POST</td>
<td>0.923</td>
</tr>
</tbody>
</table>

Table 6: Similarity between human and generated sentences before (PRE) and after (POST) observation of GEN examples

**Influence of Generated Examples** Table 8 gives some examples of judgment groups where semantic influence can be qualitatively observed in the POST sentence. The GEN example with the most influence is shown (i.e. the one most similar to the POST sentence), and we comment on the subjective evidence of their similarity. These results encourage future opportunities for explaining the exact mechanism underlying semantic influence. We discuss this further in the next section.

<table>
<thead>
<tr>
<th>Judgment</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Preferred</td>
<td>0.922</td>
</tr>
<tr>
<td>Preferred</td>
<td>0.925</td>
</tr>
</tbody>
</table>

Table 7: Similarity between POST and GEN sentences (i.e. degree of semantic influence) according to storiability preferences

\textsuperscript{10}The same core model used for computing probability scores to determine prompt difficulty, as described earlier. Here, we use the raw hidden states of the model for feature representation instead of the Masked LM probability outputs.
<table>
<thead>
<tr>
<th>Prompt</th>
<th>PRE Sentence</th>
<th>POST Sentence</th>
<th>Influential GEN Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>shoulders waves color</td>
<td>My shoulders were aching but I was set on diving through the waves, the color of the water getting deeper the further out I went.</td>
<td>Her new hair cut had the length to the shoulders, with waves of a bright pink color all the way down.</td>
<td>His hair was cropped short, flowing down his shoulders, but there were waves of the same color.</td>
<td>Connected prompt words via semantic category of hair</td>
</tr>
<tr>
<td>there die capacity</td>
<td>The bouncer thought there was a chance people might die if there was a fire because the club was way over its capacity.</td>
<td>There is no chance you’re not going to die, so you have to come to terms with that in some capacity.</td>
<td>It seems, that there is a good chance that I will die in my capacity to forgive and to get on with my life.</td>
<td>Used less literal sense of word “capacity”</td>
</tr>
<tr>
<td>meant said store</td>
<td>The child yelled at her sister not understanding what she meant when she said to her that she wanted some comics from the store.</td>
<td>It meant a lot to me when she said she was going to the toy store to get me a game.</td>
<td>It meant a lot to me, because I’d said I’d drop by the store.</td>
<td>Used phrase “it meant a lot to me”</td>
</tr>
<tr>
<td>spent wind him</td>
<td>After his run he stood by the beach, spent, as the wind whipped by him.</td>
<td>She spent the day by the water, the wind whipping her hair, aching for him.</td>
<td>She spent the rest of the day in the saddle, keeping the wind from blowing through her hair and reminding her of her promise to get him a hot bath.</td>
<td>Used expanded form of phrase “spent the day” (“spent the rest of the day”)</td>
</tr>
<tr>
<td>peculiar rob more</td>
<td>I have a peculiar friend named Rob who always wants more excitement.</td>
<td>I felt it was very peculiar that after talking to Rob for only about an hour, I wanted to know more about him.</td>
<td>They felt a peculiar attraction to Rob, but couldn’t afford to spend much more time together.</td>
<td>Referred to curiosity about Rob</td>
</tr>
</tbody>
</table>

Table 8: Examples of POST sentences demonstrating semantic influence, with subjective descriptions of how influence is seen. For reference, the PRE sentence without semantic influence is also shown.

**Discussion**

Observing automatically generated examples of sentence infilling influenced authors to better perform this infilling task on their own. Even though this is a contrived exercise different from conventional forms of creative writing, it still calls upon the same linguistic creativity. A related task is reflected in the real world through popular word games where people produce sentences given word constraints and players rate the interpretability and creativity of the resulting sentences (e.g. Cooper and McNeill, 2005). The task is also applicable to CSTs for writing: for example, a writer might want to brainstorm about potential connections between words they already have in mind, which could be facilitated by a model related to infilling. In contrast to other research on CSTs, this paper focuses less on the interactive capabilities of such systems, like enabling author control over generated output, but our findings are still relevant to interactive applications.

We chose to emphasize the authoring objective of storiability because of our focus on AI-augmented story writing. Storiability is not a one-size-fits-all metric for this research. The quality of a story can be judged on multiple dimensions that are often not consistently defined across different studies, as discussed in Celikyilmaz, Clark, and Gao (2020). Evaluations tend to target both the sensibility of stories (e.g. grammaticality, coherence, plausibility) and their more “creative” aspects (e.g. interestingness, suspensefulness, humorousness). The notion of storiability is more related to the latter group, but does not preclude other dimensions. For example, if a sentence contains grammatical errors, a person may not prefer to read the story associated with that sentence. By operationalizing storiability according to a specific question (“which sentence makes you want to read more?”), we tried to elicit judgments that encompass many ways this objective can be achieved. Future research can examine more specific formulations of this question.

An intriguing finding was the difference in outcomes according to prompt difficulty, such that only sentences for hard prompts displayed more storiability as an effect of observing generated text, with no such pattern for easy items. This points to a broad direction for future work: to examine how the demands of the writing task itself affect authors’ interaction with an automated model. For instance, authors’ engagement with writing assistance tools varies at different times during a single writing session, as discussed in Huang, Huang, and Huang (2020). This may be due to some parts of the text being harder to write than others, as hinted by the mediating effect of difficulty in our results. Interestingly, a follow-up analysis showed there were no significant differences in POST similarity to GEN examples based on difficulty, meaning that the GEN sentences for easy prompts had just as much semantic influence as those for hard prompts. Thus, this influence was somehow not as helpful in promoting storiability in the easy case. One possibility is that authors were already good at producing storiable sentences for
easy prompts in the PRE stage, so even when they were influenced by the GEN examples, this influence did not additionally benefit the POST sentences. The hard prompts may have been more challenging, giving the GEN examples a larger opportunity to enhance the POST sentences in this case. Because our evaluation did not include pairwise comparisons between sentences for easy and hard prompts, it will require further research to better understand this finding.

Our analysis of semantic influence confirms authors derived certain content from the observed examples. More investigation is needed to understand what type of content was most influential. Authors may have extracted specific words and phrases, as indicated by some of the examples in Table 8, but they did not simply copy or mimic the examples at large; if they had, there would not be a significant difference in storiability between the POST and GEN sentences as reported in Table 3. One thought is that authors utilized an idea conveyed by a GEN sentence, but reformulated the sentence to repair inadequacies such as ill-formed, awkward, or vague wording. It is also possible that the GEN examples revealed a semantic dimension by which the prompt words were related, one that authors did not initially consider in the PRE condition. The first example in Table 8 might convey this: the GEN example connects the prompt words “shoulders”, “waves”, and “color” through the conceptual dimension of “hair”. Perhaps the example triggered the author to recall this particular concept unifying the prompt words, and they emulated it in the POST sentence. One targeted metric for examining influence could focus specifically on modeling this activation of “latent” concepts. Our work quantified influence according to a single measure, but future work could attempt to narrow down the influence of specific linguistic features such as syntactic style (e.g. relative proportion of nouns, verbs, prepositions, etc.), emotional tone (e.g. joyful, sorrowful, fearful), and narrative perspective (e.g. references to pronouns and proper nouns). Existing work has addressed this by examining the strategies authors develop for eliciting precise types of content from generation models; for example, by triggering the model at certain syntactic positions in a sentence (Calderwood et al., 2020). We can use these analyses to guide future systems towards producing content authors find most helpful.

Conclusion

In this paper, we explore the question of how automatically generated text can influence human creative writing. We specifically assessed this question through the authoring task and objective of sentence infilling and storiability, respectively. In accordance with a proposed inspiration-through-observation paradigm by which automated models provide helpful examples of how to fulfill the task, we found that observing generated sentences enhanced reader-judged appeal of human-authored sentences. Our results provide empirical evidence that automated models can intervene in the writing process without necessarily replacing human effort. This invites further exploration of this paradigm for other authoring tasks and objectives. The outcome has the potential to transcend the standard of both human and computer authoring when each function independently.

References


Calderwood, A.; Qiu, V.; Gero, K. I.; and Chilton, L. B. 2020. How novelists use generative language models: An exploratory user study. In HAI-GEN+ user2agent@ IUI.


Exploring a Masked Language Model for Creative Text Transformation

Hugo Gonçalo Oliveira
CISUC, Department of Informatics Engineering
University of Coimbra, Portugal
hroliv@dei.uc.pt

Abstract
We explore a masked-language model based on BERT for shifting the meaning of text towards a target theme. Content words in the original text are masked and the model provides a list of filling candidates, out of which one is selected based on its similarity to the theme and constraints regarding morphology and metre. Experimentation is performed with Portuguese song lyrics and trade-offs between grammaticality, semantics, form, and novelty are analysed. We confirm that BERT is a useful tool for creative text transformation.

Introduction
Many creative systems rely on some kind of inspiration set, but only some assume that at transforming a single existing artefact with a pre-defined purpose. When it comes to linguistic creativity, transformation-based approaches have been adopted for the generation of song lyrics (Bay, Bodily, and Ventura, 2017; Gonçalo Oliveira, 2020) and headlines (van Stegeren and Theune, 2019; Mendes and Gonçalo Oliveira, 2020). In this domain, transformation consists of replacing parts of the text, often words, with others that meet desired constraints. Applications include advertising campaigns and others, such as the creation of parodies, either celebrating an event/someone, or making fun of them. And starting with a well-known text or melody makes the result more memorable, also helping to spread the message.

Following recent trends on using neural language models in natural language processing (NLP) and generation tasks, we explore the language-modelling capabilities of BERT (Devlin et al., 2019) for transforming creative text. BERT is a bidirectional language model based on a Transformer neural network, trained for the prediction of masked words, considering both their left and their right context. Given some text, ideally well-known, and a theme, we rely on BERT for providing replacement candidates for some words. Yet, instead of always using the first candidate, in an attempt to control the semantics of the resulting text, we consider the similarity between the resulting sequences and the theme. This is the logic underlying Zorro, a creative system that transforms Portuguese text according to a given theme. It is applied to song lyrics and, besides semantic similarity, two constraints are tested, for better consistency and aesthetics, namely morphology and metre.

As it happens for other creative artefacts, several subjective aspects must be considered during evaluation. Therefore, we had to rely on human opinions for the manual evaluation of grammaticality, semantic coherence and overall appreciation. Whenever possible, we also explored automatic and semi-automatic procedures, namely for assessing: relatedness to the theme, given by the semantic similarity between human-given titles and the original theme; novelty towards the original lyrics, given by the textual overlap between the new and the original lyrics.

Aforementioned aspects are compared for a set of lyrics transformed with different constraints, confirming some initial expectations. Results of human evaluation confirm the subjectivity of the task and, overall, show moderate success. While it is true that the meaning of the lyrics shifts towards the selected themes, trade-offs exist, and increasing the similarity with the theme, by considering more candidates, often results in more grammatical issues. Moreover, overall appreciation is lower than for the original song lyrics. Nevertheless, we believe to have confirmed that BERT is not only a powerful model for NLP, but has also potential for Computational Creativity.

In the remainder of the paper, we overview previous work on the generation of creative text with neural language models, as well as on the transformation of text. We then describe the proposed approach, followed by the experimentation setup, including implementation details. Before concluding, evaluation results are presented and discussed, together with insights and examples.

Related Work
In the last 20 years, poetry generation, including song lyrics, has arguably been one of the most active research topics in Computational Creativity, with a broad range of computational techniques explored (Gonçalo Oliveira, 2017; Lamb, Brown, and Clarke, 2017). Lately, we have seen a growing application of models based on neural networks for this task. For instance, recurrent neural networks with LSTM layers were used for generating rap lyrics (Potash, Romanov, and Rumshisky, 2015). An advantage of such approaches is that they do not require handcrafted rules.

Specifically, the introduction of Transformers (Vaswani et al., 2017) made possible the development of large language models, like GPT (Radford et al., 2019), with powerful gen-
eration capabilities. This model was used for poetry generation, namely Chinese classic poetry (Liao et al., 2019), after pre-training on a corpus of this kind of text. Using only poems with a specific format, GPT could be fine-tuned for that format, characterised by the number of lines, their metre and rhymes. However, the latter features are only captured implicitly. So, unless specific symbols are introduced for length, rhyme and others (Li et al., 2020), not all formal constraints are guaranteed to be met.

Even if previous approaches may result in fluent text that matches desired formal and aesthetic requirements, they give few or no control on the message to transmit. To deal with this limitation, a Transformer-based autoencoder can learn to map input words to related text, while matching the target form (Nikolov et al., 2020). Content words can be stripped from original lyrics, and the model trained to reconstruct the latter from part of the former and their synonyms. Generating lyrics on a new topic becomes a matter of changing the input words to others related to the target topic.

While the previous work generates new text from scratch, others assume the goal of transforming a given text with some creative purpose. Here, another Transformer-based language model, BERT (Devlin et al., 2019), can be useful for suggesting replacements. BERT is less used in generation tasks, but it is very powerful when it comes to predicting words based on their context. A BERT model is often pre-trained in two tasks, Masked Language Model (MLM) and Next Sentence Prediction, but can be fine-tuned for many downstream tasks, like Question Answering or Natural Language Inference. In the MLM task, about 15% of the tokens in the training sequences are masked (i.e., replaced by the token [MASK]) and the model is trained to predict each, considering both their left and right context. This makes such a model different from conditional language models.

In fact, the work of Nikolov et al. (2020) has a post-processing step where words ending each line are masked, and BERT used for providing suitable replacements, out of which those maximising rhymes are selected.

Though not using neural networks, several works have tackled the transformation of text. This approach has been applied to the automatic generation of poetry (Toivanen et al., 2012), also including song lyrics, where words have been replaced by: key concepts extracted from daily news, always considering syntactic and metre constraints (Gatti et al., 2017); words selected according to a theme, emotion, meter, or rhyme (Bay, Bodily, and Ventura, 2017); words whose relation to a new theme is analogous to the relation between the original words and the original song title (Gonçalo Oliveira, 2020). In the last two works, distributional representations of words (word embeddings) were used for computing semantic similarity.

Moreover, transformation-based approaches were applied to the adaptation of human-produced news headlines to new contexts, by replacing the nouns in the headlines with nouns from the context (van Stegeren and Theune, 2019), or replacing pairs of content words, based on analogy (Mendes and Gonçalo Oliveira, 2020).

In our work, we combine the earlier approach with the previous constraints for text transformation, i.e., we use BERT for suggesting a large list of replacement candidates, and further constrain the selection, based on a target theme, as well as morphology and metre constraints.

**Approach**

We exploit the capability of BERT predicting masked words in the transformation of text. For this purpose, we mask the content words in the text we want to transform and use BERT for obtaining replacement candidates. The first candidate will be BERT’s best prediction and will result in the most fluent sentence. On the other hand, the result will always be the same and, except for the words to replace, we will not have any control in the result.

But BERT can provide a ranked list of replacement candidates. So, we can explore this list and select the candidate that best suits our goal, in this case, shifting the meaning of the text towards a given theme. Moreover, other constraints can be applied to the candidates, namely on morphology and on their metre. We consider four configurations:

- **Basic**: selects the candidate that maximises the semantic similarity with the theme.
- **Morphology**: besides similarity, for increasing syntactic consistency, only considers candidates that may have the same part-of-speech (PoS) and are inflected (number, gender, tense) as the replaced word.
- **Metre**: besides similarity, to agree with the original metre, only considers candidates with the same number of syllables and stress position as the replaced word. For line-ending words, it gives priority to candidates that rhyme with the original word.
- **Morphology and Metre**: combines both of the previous (hereafter, MM).

Constraints are applied to any list of candidates, and only nouns, verbs and adjectives in the original text are masked. This is done both for keeping some syntactic consistency and resemblance with the original text, but also because similarity between function words (e.g., prepositions and determiners) and any other would be meaningless. We should add that, even though the method tries to replace all content words, it might not find a replacement that matches all the morphology and metre constraints. Only in that case, the original word is kept.

BERT can also be used for computing the semantic similarity. We use it for representing both the theme and the resulting sequences, after each replacement, and then compute their cosine. The higher the cosine, the higher the similarity.

**Experimentation Setup**

Even though the proposed approach is applicable to any language, we adopted it in the development of Zorro, a creative system for the transformation of Portuguese text, ideally poems. This section describes its implementation and illustrates the process with an example.
Implementation

Implementation of Zorro is based on Python and relies on the transformers library from Hugging Face, for loading BERT; and on NLTK (Loper and Bird, 2002), for PoS tagging. As our language model, we use BERTimbaú (Souza, Nogueira, and Lotufo, 2020) base, a pre-trained BERT model for Portuguese, with 12 layers, that encodes text sequences in 768-sized vectors. In addition to NLTK, LABEL-Lex (Ranchhod, Mota, and Baptista, 1999), a morphology lexicon for Portuguese, is used for identifying content words and checking the PoS of the candidates. Syllable division and rhyme identification were performed according to a set of rules adapted from Tra-la-Lyrics (Gonçalo Oliveira, Cardoso, and Pereira, 2007).

Example

To illustrate the proposed approach, we consider the lyrics of the song Efectivamente (in English, ‘actually’), by the Portuguese band GNR. This song starts with the following four lines, roughly translated by those following them:

Adoro o campo as árvores e as flores
Jarros e perpetuos amores
Que fiquem perto da esplanada de um bar
Pássaros estúpidos a esvoar

I love the countryside, the trees and the flowers
Arums and perpetual loves
That stay close to the terrace of a bar
Stupid birds flying

In order to transform these lyrics according to a theme, each line is first PoS-tagged. Then, for each content word:

1. It is replaced by the mask token [MASK];
2. A ranked list of filling candidates is obtained from BERT;
3. Similarity between the theme and the sequence resulting from filling the mask with each candidate is computed. If it is the highest so far, it is kept.

For the theme criatividade computacional ('computational creativity'), and considering the first 500 filling candidates, with no additional constraint, here is how the previous steps would be instantiated for the first content word (in English, ‘I’ love):

1. Masked sequence:
   
   [MASK] o campo as árvores e as flores

2. List of candidates:
   
   [Sobre, E, Entre, E, Para, São, ... ]
   ('about', 'and', 'between', 'is', 'for', 'are')

3. Maximum similarity with criatividade computacional:
   
   Usingo('using')

   Selected replacement: Usando

   Then, for the second content word ('country'):

1. Masked sequence:
   
   Usando o [MASK] as árvores e as flores

   'software' and 'modifying'). But this was done at the expense of less syntactic consistency and stranger words.
e.g.: in the first line, the word *software* is used as a verb; in the second, an unknown word, *Ino*, possibly a suffix, is used; in the fourth line, there is a number inconsistency, as *máquinas* (‘machines’) is in the plural, but its modifier *continua* (‘continuous’) is in the singular. While the first candidates should suit the syntactic structure well, matching the PoS and inflection, the lower we get in the rank, more and more words for which this does not happen will appear. This is especially true for poetry, where less common syntactic structures are frequent and some sentences are broken into different lines.

A way to minimise the latter issues would be constraining the candidates to only those that match both the PoS and the inflection of the original words they will replace. This is done with the help of the lexicon, which should have all the possible PoS of each candidate. Adding this constrain to the previous configuration results in the following text:

```
desenho o design as máquinas e as cores
robôs e inúmeros rumores
Que contem perto da teoria de um ar
cálculos cenários a utilizar

drawing the design, the machines and the colours
robots and countless rumours
Which count close to the theory of an air
scenarios calculations to be used
```

Not only it uses words in the domain of the theme, but it follows the original rhythm, has two rhymes, and no syntactic issues, except for an odd last line. Semantic coherence could also be better, especially for the third line.

For illustrating purposes, we show the result of the last configuration in the same lyrics, but now with a different theme, *liga dos campeões* (Champions League):

```
conjunto o clube as líderes e as cores
Clubes e teóricos cantores
Que tenham perto da temporada de um mar
títulos estádios a classificar

set the club, the leaders and the colours
Singing clubs and theorists
That have close to the season of a sea
stadium titles to be classified
```

**Evaluation**

Experimentation showed interesting results regarding syntax, semantics and metre, all important in a poem. However, these are all subjective aspects, for which we cannot rely exclusively on the opinion of a single person, especially if they were involved in the development of the system.

So, a sample of lyrics was created for the evaluation of the proposed approach. We report on its assessment, starting with a human evaluation, but also applying automatic measures for computing the similarity and novelty.

**Data Sample**

Results were produced for evaluation purposes, with:

- Lyrics of **five** well-known Portuguese pop-rock songs: *Efetivamente*, by GNR; *Contenêntes* (containers), by Xutos & Pontapés; *Estou Além* (I’m beyond), by António Variações; *O Anzol* (the fishing hook), by Rádio Macau; *Cavalos de Corrida* (racehorses), by UHF.

- Using **five** different themes, namely *criatividade computacional* (computational creativity), plus four trending topics: *portal das finanças* (the website where Portuguese contributors declare their taxes), *liga dos campeões* (Champions League), *festival da eurovisão* (Eurovision Song Contest), *plano de vacinação* (vaccination plan).

- Following the **four** different strategies: Basic, Morphology, Metre, MM.

- Considering the top-500 (more conservative) and the top-5000 first replacement candidates.

This results in 200 different combinations (*5* × *5* × *4* × *2*) and thus 200 different new lyrics.

**Human Evaluation**

We resorted to the crowdsourcing platform Amazon Mechanical Turk (AMT), where human workers were instructed to perform a task consisting of: (1) reading a presented poem (i.e., one of the lyrics in the sample); (2) using Likert scales for rating it according to three main aspects (grammaticality, semantic coherence, overall appreciation) and providing a suitable title in a text field. Instructions and questions were written in Portuguese, the same language as the text. The translation of the questions was:

1. At the grammar level, the text has: 1-Many issues, 3-Some issues, 5-No issues.
2. On the transmitted message, the text: 1-Does not make any sense; 3-Has some coherence; 5-Is perfectly coherent.
3. Considering its contents, a good title for the poem is: ...
4. My overall appreciation of the poem was: 1-Hated it; 3-Interesting; 5-Loved it.

For each of the 200 lyrics, questions were answered by two different workers, who were not told that the text had been automatically transformed. Given titles are important for assessing meaningfulness, i.e., the more semantically similar they are to the themes, the better the theme is expressed by the lyrics.

Initially, it was also our intention to gather human opinions on two additional aspects of the new lyrics: rhythm and novelty, both considering the original lyrics. Yet, for this to work well, the workers had to previously know the melody of the original songs and their lyrics. All songs are by Portuguese artists, and even if they are popular enough for being known by a vast Portuguese audience, their popularity is mostly restricted to Portugal and Portuguese people. This meant that we would have to restrict the evaluation to workers located in Portugal. We actually started to do it, but had no answers for some time, which made us open the survey to other Portuguese-speaking countries, namely Brazil. For this reason, it is expected that workers will not be familiar with the original lyrics and thus unapt for as-
sessing the rhythm and novelty of the new lyrics. While an approximation to the latter was obtained automatically (see Novelty below), we ended up not assessing the rhythm. Still, from our observation of the results, we can assume that, for a large majority of cases, the Metre constraint guarantees that all replacements have the same metre as the original words they are replacing. On the other hand, this is not always the case of rhymes. We now look at the results for each human-assessed aspect.

**Grammaticality** Table 1 summarises the scores regarding the grammaticality of the lyrics transformed by each configuration with a different number of top candidates considered (Top-k). It shows the proportion of scores below, equal and above 3, followed by the median score. The latter suggests that, except perhaps for the Morphology and Metre configuration with 500 candidates (hereafter, MM-500), there are no substantial differences among configurations. Yet, even if differences are low, score distribution is in line with three expectations: grammaticality is harmed by a higher number of candidates and by the metre constraint; with the morphology constraint on and 500 candidates, the proportion of lyrics with scores > 3 is higher, even when the metre constraint is added. In fact, the highest proportion (62%) is when both constraints are used. Here, we speculate that a correct metre contributes to better reading, possibly hiding grammar issues. This, however, does not seem to make a difference for the Basic configuration, as the proportion is the same for the three categories (< 3, 3, > 3), either considering 500 or 5,000 candidates.

<table>
<thead>
<tr>
<th>Config</th>
<th>Top-k</th>
<th>&lt; 3</th>
<th>3</th>
<th>&gt; 3</th>
<th>Med</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>500</td>
<td>24%</td>
<td>40%</td>
<td>36%</td>
<td>3.0</td>
</tr>
<tr>
<td>Basic</td>
<td>5000</td>
<td>24%</td>
<td>40%</td>
<td>36%</td>
<td>3.0</td>
</tr>
<tr>
<td>Morphology</td>
<td>500</td>
<td>16%</td>
<td>38%</td>
<td>46%</td>
<td>3.0</td>
</tr>
<tr>
<td>Morphology</td>
<td>5000</td>
<td>28%</td>
<td>42%</td>
<td>30%</td>
<td>3.0</td>
</tr>
<tr>
<td>Metre</td>
<td>500</td>
<td>34%</td>
<td>28%</td>
<td>38%</td>
<td>3.0</td>
</tr>
<tr>
<td>Metre</td>
<td>5000</td>
<td>46%</td>
<td>30%</td>
<td>24%</td>
<td>3.0</td>
</tr>
<tr>
<td>Morph+Metre</td>
<td>500</td>
<td>22%</td>
<td>16%</td>
<td>62%</td>
<td>4.0</td>
</tr>
<tr>
<td>Morph+Metre</td>
<td>5000</td>
<td>38%</td>
<td>28%</td>
<td>34%</td>
<td>3.0</td>
</tr>
</tbody>
</table>

Table 1: Grammaticality in lyrics transformed by each configuration, considering a different number of top candidates.

**Semantics** Table 2 is on the semantic-related aspects. It has the scores given by the workers for semantic coherence, followed by two similarity values which are average cosines between the BERT representations of the worker-given titles, respectively with the theme of the transformed lyrics and with the original title of the song. Since it is virtually impossible for the workers to guess exactly the words of the theme, computing the semantic similarity is a shortcut for quantifying the proximity between the meaning of the title and the theme. BERT encodings are suitable for this, as they can represent whole sequences in a vector of real numbers, based on the meaning of the words in context.

Medians are not very different than for grammaticality, i.e., combining morphology and metre with the top-500 candidates gets slightly higher than the other configurations. Scores also suggest that morphology constraints contribute to semantic coherence more than the metre.

However, we were also assessing to what extent the lyrics are related to the theme, i.e., their semantic similarity with the theme. Since there is not a threshold for which we can consider that two texts are similar enough, we look at the similarity values relatively. We can compare average similarities of the given title with the theme, for different configurations, and also with the original title of the song. As we were trying to shift the meaning of the lyrics, similarity of the given title should be higher for the theme than for the original title.

Even though average similarity values are all very close, the previous expectation is confirmed for every configuration. However, some differences are very low and not significant when the standard deviation is considered. This happens especially for the MM configuration, the one with highest scores for grammaticality and semantic coherence. In addition to what was said about this configuration, we started to question whether the application of both constraints was preventing the replacement of all words, leaving large sequences of the original lyrics, and thus a better grammaticality, semantic coherence and a meaning that is also closer to the original. In the following sections, we show examples of lyrics transformed with this configuration and compute their overlap with the original lyrics, to confirm that this is not exactly the case.

Out of curiosity, we looked at the most given titles to find that a total of nine lyrics got the title “Agora” (now). Though not very related to any of our themes, it is the starting word of all lines in the lyrics of Cavaleiros de Corrida and, as an adverb, it is never replaced. A similar situation occurred for the title “Debaixo do Sol” (under the sun), given four times, for being the end of the lyrics of O Anzol, and kept intact with metre constraints on. Other common titles that are related to the themes include: Prevenção (prevention), Virus (virus) and Gripe (flu), all related to vaccination plan, respectively given six, five and four times; “Futebol” (football), related to Champions League, given five times; and Finanças (finances), part of one of the themes, given four times.

There was a single case for which the given title was exactly the same as the theme and it was Liga dos Campeões, given to the lyrics of O Anzol, transformed with Basic-500. Other titles highly similar (cosine > 0.87) to the themes were: Os Clubes Campeões (champion clubs) and Campeão dos Campeões (champion of the champions), respectively for the lyrics of Efectivamente transformed with Basic-500, and Cavaleiros de Corrida with Basic-5000; Inovação Computacional (computational innovation) and Design de Inteligência (design of intelligence), respectively for the lyrics of Contentores transformed by Basic-5000 and Cavaleiros de Corrida with Morph-5000. This analysis showed that, for some original lyrics and themes (computational creativity,
champions league) given titles were closer to the theme than for others.

**Overall Appreciation**  Table 3 is on the overall appreciation, which, in the end, is probably what matters the most. In this aspect, we cannot say that the results were very positive, as all configurations have a great proportion of lyrics with scores 1 and 2. When comparing configurations, we see that the metre constraints alone, which favor words based on their length and stress, lead to the worst appreciation. Sometimes, a better rhythm and sound can compensate for some ungrammatical text. Yet, we recall that the workers did not know the original song nor its rhythm, meaning that metre would hardly play a role in their judgement.

<table>
<thead>
<tr>
<th>Config</th>
<th>Top-k</th>
<th>&lt;3</th>
<th>3</th>
<th>&gt;3</th>
<th>Med</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>500</td>
<td>28%</td>
<td>42%</td>
<td>30%</td>
<td>3.0</td>
</tr>
<tr>
<td>Basic</td>
<td>5000</td>
<td>34%</td>
<td>42%</td>
<td>24%</td>
<td>3.0</td>
</tr>
<tr>
<td>Morphology</td>
<td>500</td>
<td>20%</td>
<td>38%</td>
<td>42%</td>
<td>3.0</td>
</tr>
<tr>
<td>Morphology</td>
<td>5000</td>
<td>20%</td>
<td>44%</td>
<td>36%</td>
<td>3.0</td>
</tr>
<tr>
<td>Metre</td>
<td>500</td>
<td>38%</td>
<td>36%</td>
<td>26%</td>
<td>3.0</td>
</tr>
<tr>
<td>Metre</td>
<td>5000</td>
<td>40%</td>
<td>46%</td>
<td>14%</td>
<td>3.0</td>
</tr>
<tr>
<td>Morph+Metre</td>
<td>500</td>
<td>20%</td>
<td>30%</td>
<td>50%</td>
<td>3.5</td>
</tr>
<tr>
<td>Morph+Metre</td>
<td>5000</td>
<td>42%</td>
<td>34%</td>
<td>24%</td>
<td>3.0</td>
</tr>
</tbody>
</table>

Table 3: Overall appreciation for each configuration.

MM-500 and five with MM-5000, three of them in Figure 2. The remaining were by Basic-500 (1), Morph-500 (1), Metre-500 (4) and Metre-5000 (2). Semantic coherence is lower in all three examples shown. In the first and third, we note a higher presence of odd line constructions and words (e.g., *els, log, cm*).

**Novelty**

The higher average similarity between given titles and themes was already a hint that some original words were being replaced, resulting in a meaning shift. Still, we attempted to quantify the novelty, based on the overlap between the original and the transformed lyrics. Some overlap is expected, and even desired for increased familiarity, but the resulting lyrics should not just be a copy of the original.

In order to approximate novelty towards the original, we adopted ROUGE (Lin, 2004), a set of metrics commonly used for assessing automatic summarisation and machine translation, based on the overlap between the generated and the original text. The main difference here is that we are aiming for lower ROUGE scores, meaning that overlap with the original text is also lower. In the scope of Computational Creativity, ROUGE has previously been used for assessing variation in automatically-generated poems (Gonçalo Oliveira et al., 2017).

We consider the overlap between uni (R-1), bi (R-2), and tri (R-3) grams, as well as the longest common subsequence (R-LCS). Alone, the obtained values are probably not so meaningful, but they enable us to compare different configurations. For each metric considered, Table 4 shows the average values for the simplest configuration (Basic-500) and the difference for all the others. First of all, values confirm that, for any configuration, several changes are made to the original lyrics. Otherwise, they would be 1 or close. They further confirm some initial expectations, even if by low margins. For instance, novelty is higher for the Basic strategy, which does not constrain the replacements, meaning that all content words end up being replaced. It is also higher with the morphology than with the metre constraints alone, showing that it is more difficult to find words with a certain number of syllables and stress than with a target PoS and inflection. In fact, when applying only the morphology constraints, ROUGE values are almost the same as for the Basic configuration. A final confirmation is that novelty is higher when considering the top-5000 candidates, for which chances of finding suitable replacements increase.

<table>
<thead>
<tr>
<th>Config</th>
<th>Top-k</th>
<th>&lt;3</th>
<th>3</th>
<th>&gt;3</th>
<th>Med</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>500</td>
<td>28%</td>
<td>42%</td>
<td>30%</td>
<td>3.0</td>
</tr>
<tr>
<td>Basic</td>
<td>5000</td>
<td>34%</td>
<td>42%</td>
<td>24%</td>
<td>3.0</td>
</tr>
<tr>
<td>Morphology</td>
<td>500</td>
<td>20%</td>
<td>38%</td>
<td>42%</td>
<td>3.0</td>
</tr>
<tr>
<td>Morphology</td>
<td>5000</td>
<td>20%</td>
<td>44%</td>
<td>36%</td>
<td>3.0</td>
</tr>
<tr>
<td>Metre</td>
<td>500</td>
<td>38%</td>
<td>36%</td>
<td>26%</td>
<td>3.0</td>
</tr>
<tr>
<td>Metre</td>
<td>5000</td>
<td>40%</td>
<td>46%</td>
<td>14%</td>
<td>3.0</td>
</tr>
<tr>
<td>Morph+Metre</td>
<td>500</td>
<td>20%</td>
<td>30%</td>
<td>50%</td>
<td>3.5</td>
</tr>
<tr>
<td>Morph+Metre</td>
<td>5000</td>
<td>42%</td>
<td>34%</td>
<td>24%</td>
<td>3.0</td>
</tr>
</tbody>
</table>

Table 2: Semantic coherence and average similarity of given titles with the theme and the original title.
Table 4: ROUGE scores of the Basic-500 configuration and differences of the others when compared to it.

Table 5 shows the scores obtained by the original lyrics of the songs used in this experimentation. They are based on four human opinions for each of the five original lyrics. The

Original: Contentores
Theme: festival da eurovisão
Titles: a despedida (the farewell, 0.72), adeus romantico (romantic goodbye, 0.69)
Gram: 4, Sem: 4

Table 5: Scores obtained by the original lyrics of the songs used in this experimentation.

Figure 1: Lyric with high overall appreciation, all with MM-500 configuration.

Figure 2: Lyrics with low overall appreciation.

Scoring the original lyrics

Since workers were not from Portugal, they would not recognise the original lyrics. Therefore, we thought that it would be interesting to mix the original lyrics among the automatically transformed. This enabled us to gather opinions for the former, on the same aspects we were considering for the latter, also giving us another term of comparison.

Table 5 shows the scores obtained by the original lyrics of the songs used in this experimentation. They are based on four human opinions for each of the five original lyrics. The

Table 4: ROUGE scores of the Basic-500 configuration and differences of the others when compared to it.
semantic similarity of the given title was computed against the original title of the song.
Results are interesting to show that, according to human opinions, original songs still have issues. This happens for grammaticality, less expected, but also for semantics and, especially, for the overall appreciation, which is also the most subjective. Our interpretation is that song lyrics are not always easy to interpret, also due to the presence of figural language. Together with the position of the line breaks, this might also have a minor impact on grammaticality. Not to mention that all lyrics were in European Portuguese, but scored by Brazilian workers.

Regarding the average similarity between given and original titles, it is only higher than similarity between given titles for the MM configuration and their themes. This suggests that, even though replacements are made in this configuration, relatedness between the target themes and the transmitted message could be stronger.

Conclusion
We have explored the application of a popular masked language model, BERT, in the transformation of creative text, namely Portuguese song lyrics. We took advantage of the mask-filling feature of this model and further constrained the filling predictions, according to their similarity to a given theme, PoS and metre.

There are trade-offs between increasing the similarity with a theme and keeping the text grammatical, so, finding the right balance can be tricky. Our evaluation was limited to a language, a set of original lyrics and constraints, in any case leading to an overall appreciation of the transformed lyrics below the one for the original. Towards more successful results, testing with different parameters is required, and, depending on the final purpose, human curation might still be necessary in the end. A possibly erroneous conclusion was that matching the metre does not contribute to higher appreciation, but this might be due to our impediments on properly assessing singability, due to lack of workers that knew the original songs.

We cannot say that we are completely satisfied, but interesting results were obtained. Enough to believe that BERT should be seen as a useful tool in the transformation of creative text. For stronger conclusions, further experimentation is needed, e.g., in other languages, considering other texts, different numbers of candidates and possibly other constraints. This also applies to alternative ways of using BERT, e.g., on computing the similarity of each candidate to the theme (e.g., just the token, right context followed by the token, full resulting sequence), or of getting sequence representations (e.g., from different layers, considering the contextual token embeddings), possibly after fine-tuning the model to some task where creative text is used.

Acknowledgments
This work was supported by national funds through the FCT – Foundation for Science and Technology, I.P., within the scope of the project CISUC – UID/CEC/00326/2020 and by European Social Fund, through the Regional Operational Program Centro 2020.

References
Table 5: Scores for the original lyrics.

<table>
<thead>
<tr>
<th></th>
<th>Grammar</th>
<th></th>
<th>Semantic Coherence</th>
<th></th>
<th>Appreciation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;3</td>
<td>3</td>
<td>&gt;3</td>
<td>Med</td>
<td></td>
</tr>
<tr>
<td>10%</td>
<td>5%</td>
<td>85%</td>
<td>4.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;3</td>
<td>3</td>
<td>&gt;3</td>
<td>Med</td>
<td>Sim(title)</td>
</tr>
<tr>
<td>10%</td>
<td>10%</td>
<td>80%</td>
<td>4.0</td>
<td>0.670±0.08</td>
<td></td>
</tr>
<tr>
<td>5%</td>
<td>30%</td>
<td>65%</td>
<td>4.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


Appendix

For better understanding of their contents, Figures 3 and 4 show rough English translations of the Portuguese lyrics in Figures 1 and 2, respectively.
Original: Contentores
Theme: festiva da eurovisão
Titles: a despedida (the farewell, 0.72), adeus romantico (romantic goodbye, 0.69)
Gram: 4, Sem: 4

The Greek crisis and arrival in the corridors
Farewell to my shoes cause I am going
To another world
In a star appearing in a judicial project
This where I go does not make anything bad
To the theatre world
All the colours have changed
make alone the readers
And in an impossible crisis
Seeing Central Lisbon
Do nothing wrong

Original: Contentores
Theme: portal das finanças
Titles: eu me vou (‘I’m leaving, 0.68), impostos (taxes, 0.70)
Gram: 3.5, Sem: 3

The fake and locked sheet in the corridors
Farewell to my taxes, I’m going
To another world
In a foreign bank in a commercial govern-
ment
It doesn’t save that bad this where I’m going
All over the world
All the pains have changed
They use the values alone
And in an impossible crisis
I sell the general magazine
Does not enter that bad

Original: Estou Além
Theme: criatividade computacional
Titles: ao me conhecer (meeting me, 0.68), arte de gerar (art of generating, 0.68)
Gram: 4.5, Sem: 4.5

No need to transform
This utility system
The art of generating
So that it doesn’t become late
I am not of what there is I use
Is it from of this dimension
But why does it make me abuse
Who makes me give my hand
I will continue to explore
To whom I give myself
Because until here I only
I have who, who I’ve never seen
Because I only find who
Who I haven’t met

Figure 3: Lyrics with high overall appreciation, all with MM-500 configuration.

Original: Cavalos de Corrida
Theme: festiva da eurovisão
Config: MM-500
Titles: hipocresia (hypocrisy, 0.59), evento de comida (food event, 0.59)
Gram: 3.5, Sem: 2.5

Now that Europe is over, and the
media follow in an effort
Now they’ve all played, the mythol-
yogy at stake
Now they make the media, destroy-
ing all the laws
Now they follow in the hole and
they do it at any price
Now, now, now, now, you give a
food event

Original: Contentores
Theme: plano de vacinacao
Config: MM-5000
Titles: virus (virus, 0.70), o virus (the virus, 0.70)
Gram: 3.5, Sem: 2.5

Influenza pay and consultation in the bearers
Farewell to my archives, I’m leaving
To another background
In a virus treated in an oriental scheme
It does not treat badly this where I go
To the complete fund
indicate all the pains
should alone the underaged
And in a compatible list
Home Health Plan
It does not treat that badly

Original: O Anzol
Theme: criatividade computacional
Config: MM-5000
Titles: medir (measuring, 0.80), medição (measurement, 0.80)
Gram: 2, Sem: 2

Oh, I’ve already measured
think about moving the top in log from blue
For structural cm
But only then I measured
make plural you already have, it assembles
someone
Who assembles a mental idea
I am not whether to measure
Or generate the pattern
There’s nothing new here anymore
Under the sun

Figure 4: Lyrics with low overall appreciation.
Nightmare Machine: A Large-Scale Study to Induce Fear using Artificial Intelligence

Pinar Yanardag1 Nick Obradovich2 Manuel Cebrian2 Iyad Rahwan2
1Bogazici University 2MPI for Human Development
yanardag.pinar@gmail.com {obradovich,cebrian,rahwan}@mpib-berlin.mpg.de

Abstract

As Artificial Intelligence makes strides in emulating human performance in analytical tasks, an important question surfaces: can machines induce extreme human emotions at scale? In this work, we investigate a case study, Nightmare Machine (nightmare.mit.edu) towards a particular emotion: fear. We use a deep-learning based approach that induces states of anxiety and negative affect by generating de-novo eerie images. Our system attracted the attention of hundreds of thousands of participants from 147 countries who produced over 1,000,000 evaluations of the generated images. First, we perform various exploratory data analysis tasks on the collected data in order to investigate the potential of the generated images, such as whether there exists a correlation between preferences of the participants based on geographic location. Then, we perform a validation study on n = 752 subjects to verify whether the generated images psychologically move people on psychometrically validated measures of effect and anxiety such as I-PANAS-SF (Thompson 2007) and STAI-SF (Marteau and Bekker 1992). Our experiments show that the generated images produced statistically significant increases in negative affect and state anxiety compared to the control images. We make our dataset publicly available at https://github.com/catlab-team/nighmaremachine.

Introduction

Recent advances in artificial intelligence achieved significant breakthroughs that exceed human capabilities and gained an immense amount of attention due to their success in several areas including computer vision (Voulodimos et al. 2018), language modeling (Jozeofwicz et al. 2016), and robotics (Pierson and Gashler 2017). Deep learning, a sub-field of artificial intelligence, enabled researchers to discover complex patterns in extremely large datasets and widely deployed in academia and industry (Le et al. 2020; Luckow et al. 2016). Deep learning based systems started to gain popularity when a convolutional neural network outperformed a large-scale image classification task at ImageNet Large-Scale Visual Recognition Challenge (Russakovsky et al. 2015). This success, supported by the remarkable developments of powerful processors (GPUs), and explosive growth of data, enabled the rise of deep learning. Since then, deep learning has become state-of-the-art approach for a large variety of problems, including image processing (Hemanth and Estrela 2017), natural language processing (Deng and Liu 2018), speech recognition (Deng and Yu 2014) and even defeating world’s best Go and chess players (Silver et al. 2016; Silver et al. 2017) or beats human champions in Jeopardy such as IBM’s Watson (Watson 2014). As Artificial Intelligence makes strides in solving challenging analytical problems like checkers (Samuel 1967), chess (Silver et al. 2017) or video-games (Vinyals et al. 2017) society takes solace in the implicit belief that the subset of human tasks that rely on the understanding, managing, and inducing human emotions are still far from the ability of machines to outperform humans. But are they? Can computers learn to induce emotions faster and better than humans can?

Detecting emotion can be considered as a first step towards inducing emotion. Machine learning is enjoying rapid advancement on this front (Hossain and Muhammad 2019) and initial algorithms were able to detect positive and negative emotion (Liu, Zheng, and Lu 2016). More recently, Natural Language Processing (NLP) has been able to infer not only the mood expressed in text but also irony and sarcasm (Schifanella et al. 2016) and, in some cases, humor (Chen and Soo 2018). Affective Computing, computational tools to sense and improve human-computer communication, is also enjoying a steady revival (Picard 2000). Going from the detection of emotion to the induction of emotion is, however, a big leap, and one that we tackle in this work. Can Artificial Intelligence not only detect but induce specific emotions in humans, in particular, fear? Attempts at fear induction taking the form of stories and visual images pervade the history of human culture. Creating a visceral emotion such as fear remains one of the cornerstones of human creativity. In this work, we explore a way to combine deep learning and crowd-sourcing to test whether fear can be induced at scale. To our knowledge, we are the first to automate the production of scary images. While computers can detect images that may be upsetting, there’s no previous literature on seeing whether computers can generate them. In this work, we propose a deep-learning based approach to a particular emotion; fear, and explore whether we can induce states of anxiety and negative affect with the generated images. Our platform gained wide attention from all over the world and collected over one million votes on the generated images from 147 countries.
Related Work

In this section, we first cover related work in the intersection of emotions and artificial intelligence. Then, we discuss crowd-sourced tools that utilize artificial intelligence applications. Finally, we briefly cover related work in generative models.

Emotions and Artificial Intelligence

Recent advances in deep-learning encouraged researchers to investigate the usage of artificial intelligence in terms of emotions. Most of the existing work focuses on the classification or detection of certain emotions on facial data. (Ng et al. 2015) used a convolutional neural network architecture (CNN) combined with a transfer learning approach and performed emotion recognition using EmotiW (Kahou et al. 2013), a face expression dataset that includes a wide range of emotions including happy, sad, surprised, fear, angry and disgusted. (Jain et al. 2018) tackled facial expression recognition (FER) using a hybrid convolution-recurrent neural network that extracts the relations within facial images by using the recurrent network for temporal dependencies.

Another line of work focuses on emotion recognition from speech. (Hossain and Muhammad 2019) proposed an audio-visual emotion recognition system using CNNs on an emotion dataset that consists of speech and video. They proposed to process speech signal in the frequency domain and used corresponding Mel-spectrograms as an image which is fed to a CNN. The output is fused with video signals and fed into two consecutive extreme learning machines and a support vector machine (SVM) for the classification of the emotions. (Satt, Rozenberg, and Hoory 2017) used an end-to-end deep neural network on raw spectrograms. They combined a noise reduction solution based on harmonic filtering to perform emotion recognition from speech under limited latency constraint and achieved state-of-the-art accuracy on popular benchmarking dataset IEMOACP (Busso et al. 2008).

Several studies investigated the potential of using deep learning for empathy-related applications. For instance, (Barmar 2017) used a neural network based approach to classify empathy and personal distress on facial muscle activities. They also investigated which facial muscle movements contribute the most to predict empathy. (McQuiggan et al. 2008) proposed a data-driven inductive approach for learning empathy models including both reactive and parallel empathetic expression. Their approach focuses on the observation of empathy in action and tries to understand the psychological aspects of empathetic assessment. (McQuiggan and Lester 2006) proposed a data-driven framework for extracting models of empathy that are empirically grounded from observations of human to human social interactions. (Gibson et al. 2016) used a deep neural network system for predicting empathy ratings from transcripts of counselors. To pursue this goal, they utilized a recurrent neural network that matches the transcript of a speaker with a task-specific behavioral act.

Crowd-sourced AI Tools and AI-based Creativity

The field of machine learning gained attention due to the remarkable results in several important tasks in computer vision, natural language processing and robotics areas. Recently, researchers focused on combining generative models with crowd-sourcing efforts for creative applications.

Deep Dream Generator (DeepDreamGenerator) is a computer vision tool that helps users to experiment with deep learning algorithms for creativity. Neural style transfer algorithm (Lee et al. 2018) enabled users to experiment with painting styles on any given image (DeepArt). In addition to computer vision tools, music-based platforms such as Magenta (Magenta) offers a large collection of music-based tools using a recurrent neural network based system that generates notes based on melodies provided by the users. Botnik (Botnik), GPT-2 (Radford et al. 2019a) are among the text-based platforms heavily explored for creative writing. Botnik offers a keyboard-based interface where users can collaboratively create AI-assisted text-based content. GPT-2 (Radford et al. 2019b) model uses large-scale datasets which helped users to create a variety of applications ranging from novels (GPT) to poetry (Branwen). Computationally creative Twitter bots are also utilized in several studies. (Yanardag, Cebrian, and Rahwan 2021) explores Twitter as a medium for creating horror stories in a collaborative fashion with Twitter users. (Oliveira 2017) proposes a bot that posts poems inspired by Twitter trends.

Generative Models

Generative Adversarial Networks (GANs) (Goodfellow et al. 2014a) aim to model the image space so that they generate images that are indistinguishable from those in the dataset. The adversarial part of the network detects whether the produced images are from the training dataset (or fake), and the generative part tries to create images that are similar to the dataset. DC-GAN (Radford, Metz, and Chintala 2015) is one of the first GAN models that directly extends the GAN architecture by using convolutional layers in the discriminator and convolutional-transpose layers in the generator. StyleGAN (Karras, Laine, and Aila 2018) and StyleGAN2 (Karras et al. 2020) are among popular GAN approaches that generates high-resolution images. They use a mapping network with an 8-layer multilayer perceptron (MLP) which fits input latent code onto an intermediate la-
tent space. BigGAN (Brock, Donahue, and Simonyan 2018) is another large-scale model trained on ImageNet (Rus- sakovsky et al. 2015) and utilizes the intermediate layers by taking the latent vector as input as well as a class vector that acts as a conditional information.

**Image Generation**

A two-stage architecture that combines generative adversarial networks (GANs) (Goodfellow et al. 2014b) and neural style transfer (Gatys, Ecker, and Bethge 2015) is used for generating the images used in this study. GANs are a class of neural networks that have gained popularity in recent years, with the most common application area being image generation. GANs estimate generative models with an adversarial process by simultaneously training two models: a generative model $G$ that represents the data distribution, and a discriminative model $D$ that estimates the probability of whether a sample comes from the model distribution or the data distribution. GANs train the generator $G$ and the discriminator $D$ through playing a mini-max game: $D$ maximizes the expected log-likelihood of distinguishing real samples from the fake ones, and $G$ maximizes the probability of $D$ making a mistake. The equilibrium of this game is reached when the generator is generating fakes that look like real as if they came directly from the training set, and the discriminator can not distinguish between the fake ones and real ones with a 50% confidence.

We used DC-GAN (Radford, Metz, and Chintala 2015) model that directly extends the GAN architecture by using convolutional layers in the discriminator and convolutional-transpose layers in the generator. The all convolutional net (Dosovitskiy, Tobias Springenberg, and Brox 2015) is used in its generator and discriminator which replaces the deterministic spatial pooling functions with strided convolutions and enables the network to learn its own spatial downsampling, along with batch norm layers, and LeakyReLU activations. The input to the discriminator is a $3 \times 64 \times 64$ image and output is a probability of the input belonging to the real data distribution. The input to the generator is a latent vector drawn from a prior distribution and the output is a $3 \times 64 \times 64$ image.

One approach to generating the images for our task is to...
directly train a DC-GAN model on a collection of scary images. However, there is no such data collection suitable for this purpose. Therefore, we employ a two-stage strategy (see Figure 2 for an illustration of the framework). The first step of our approach is training a DC-GAN model on a large-scale dataset called CelebFaces Attributes Dataset (CelebA) (Sun et al. 2014). The CelebA dataset contains 202,599 celebrity images with coarse alignment, each with 40 attribute annotations. We trained DC-GAN to generate face samples of $64 \times 64$ pixels. Figure 2 (denoted with Original Faces) shows a list of randomly generated faces from the trained DC-GAN model.

The second step is turning normal faces into scary images using neural style transfer (Gatys, Ecker, and Bethge 2015). Neural style transfer is an optimization method that mixes the content and style representations from two different images. The key observation the style transfer method employs is that the representations of content and style in the CNNs are separable and both representations can be manipulated in order to create new images. It synthesises a new image by simultaneously matching the content representation of an image and the style representation of a style image. It takes three inputs; a source image, a content image and a style image and uses two distance functions for optimization. The first distance function describes the difference between the content of the source and the content images, and the second distance function measures the difference between the two images in terms of their style. The objective is then to transform the source image to minimize the content distance with the content image and the style distance with the style image. We used a single style image for our experiments (see Figure 2). We fed randomly generated DC-GAN images to Neural Style Transfer model and generated a list of modified images to be used in our experiments. Figure 2 (denoted with Modified Faces) shows a selection of transformed images used in our experiments.

Exploratory Data Analysis

During a period of 9 months, we collected a high-volume dataset of emotional preferences with our crowd-sourcing platform http://nightmare.mit.edu and use it to study emotion, in particular, fear. Our dataset consist of 500 computer-generated images to vote as Scary or Not Scary. Figure 3 shows the histogram of Scary and Not Scary votes where the number of votes seems to be distributed equally with a slight shift on the right towards Scary votes. Our dataset consists of votes collected from 147 countries. Figure 4 illustrates the distribution of the collected votes over the globe. We can see that the majority of the votes are focused on United States, followed by Japan, Great Britain, Canada, and Australia.

Using this large-scale dataset, we seek to answer some of the interesting questions that can be raised as follows:

• Can we learn which images are particularly scary and distinguish between Scary and Non-Scary images?
• Are there different sub-groups within images that are labeled as scary and non-scary?
• Do participants have different preferences on what is scary and what is not?
• Is there any relationship between geographic location and preferences on the scariness?

Can we learn which images are particularly scary and distinguish between Scary and Non-Scary images? We explore whether a neural network model can learn to separate between images labeled as Scary and Not Scary. We train a convolutional neural network classifier to recognize fear by extending VGG-16 network architecture (Simonyan and Zisserman 2014). VGG-16 is a large model designed for multi-class classification, pre-trained on ImageNet dataset (Russakovsky et al. 2015) and successfully pushed the error rate to $< 10\%$ on ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014 competition (Russakovsky et al. 2015). VGG-16 network comprises 13 convolutional layers, divided in five groups. In addition, the network has 3 fully connected layers, and 5 pooling layers. The convolutional
Figure 5: Embeddings of Scary (grey clusters) and Not Scary images (colored clusters) into $d = 100$ dimensional space. Closeness in the space reflects similarity. The leftmost non-scary cluster ($CLUSTER_A$) consist of faces that are blurry and deformed while right-most non-scary cluster ($CLUSTER_B$) displays images that are dark.

layers consist of a set of kernels where each kernel is convolved with the input volume to compute hidden activations during the forward pass. Related parameters are updated using a back-propagation pass. We re-purpose VGG-16 network pre-trained on ImageNet by performing fine-tuning. In particular, we replace the final fully-connected layer of VGG-16 with two neurons corresponding to two classes, Scary and Not Scary. We curated a balanced dataset of 200 scary images and 200 non-scary images for classification. Our dataset is created as follows: we used 500 images and their corresponding votes and sorted them by the number of Scary (and respectively Not Scary) votes. We then selected the top 200 images for each category and labeled them as Scary (and respectively, Not Scary) for classification. For each class, we used 100 images for training, 30 images for validation, and 70 for testing, and used 10-fold cross-validation to evaluate the results. We obtained 65% accuracy on a balanced dataset where the baseline accuracy is 50%. This result indicates that there is some common consensus among the users about the scariness of the images, and we can distinguish between which images are Scary or Not Scary to a certain extend.

Are there different sub-groups within images that are labeled as scary and non-scary? In order to understand the relationship between Scary and Not Scary images, we built an image-embedding framework using word-embedding techniques. Word embedding methods recently gained popularity due to their success in accurately estimating the relationship between words in language models. We used Word2Vec method (Mikolov et al. 2013) where we treated each user profile as a sentence and each voted image as a word. Similar to how word-embedding methods capture the similarity of words by mining the co-occurrence relationship in a sentence, our aim is to capture similar images that users find Scary (and similarly, Not Scary). We used Word2Vec tool and embed images into $d$-dimensional space where $d = 100$. Figure 5 shows embeddings of 500 images where closeness in the space reflects similarity. As can be seen from the figure, scary images (grey clusters) and non-scary images (colored clusters) are clustered consistently. An interesting observation is that even though scary clusters are relatively close to each other, non-scary clusters are spread through the latent space. This indicates that while there is a common consensus on the groups of images that are labeled as Scary, there are particular characteristics of non-scary images that affect different users which results in many separate clusters are formed in the latent space. For instance, while the left-most non-scary cluster (labeled as $CLUSTER_A$) consists of faces that are blurry and deformed, the right-most non-scary cluster (labeled as $CLUSTER_B$) displays images that are mostly dark.

Do participants have different preferences on what is scary and what is not? We explore whether users have common preferences over the images by using a collaborative-filtering approach. Collaborative filtering is a mechanism that learns user preferences towards items by mining implicit or explicit interests a user expresses using rating information (e.g., books, movies, or products). The expressed ratings of users are matched against other users and a hidden representation of user preferences is learned. This information is then utilized to find people with the most similar preferences and to recommend items that similar users liked. A popular collaborative filtering method is matrix factorization which learns a model from incomplete rating data. We built a user-item matrix $M$ of size $m \times n$ where $m$ is number of users who visited our system, and $n$ is number of images available for users to vote. Let $i$ represent an arbitrary user who visited our system, $j$ represents an arbitrary image user $i$ voted. Then $M_{ij}$ corresponds to the rating user $i$ expressed. In particular, we interpret Scary votes as a rating of +1 and Not Scary votes as a rating of −1. If user $i$ has not been shown image $j$, then $M_{ij} = 0$ indicating an unobserved rating. The goal is then to approximate incomplete matrix $M$ by using matrix factorization. We used LIB-PMF (Yu et al. 2012), an efficient and parallelizable method for matrix factorization in large-scale rec-

Figure 6: Images on the left shows a set of images user found scary, and images on the right are the recommended images from the model.

Proceedings of the 12th International Conference on Computational Creativity (ICCC ’21)
ISBN: 978-989-54160-3-5
76
ommender systems. We curated a dataset for a randomly selected 10,000 users and 500 face images. We split the data into training and testing set with a 70% to 30% rate, respectively. In total, the training dataset contains 199,436 ratings, and testing dataset contains 85,419 ratings. The goal is then to learn user preferences on the training dataset, and then to predict the unseen ratings in the testing set. We used Root Mean Square Error (RMSE) for evaluating the results. RMSE measures the average magnitude of the error, and it is defined as the square root of the average of squared differences between predictions and actual observations. On 85,419 ratings, we obtained a RMSE of 0.93 for the distribution of actual and predicted ratings. Figure 6 shows a qualitative example of our recommender system where top images a random user liked (i.e. rated as Scary) and top recommended images that user might like (i.e. might find Scary) are shown.

Is there any relationship between geographic location and preferences on the scariness? We explore how preferences on the votes change based on geographic location, and investigate whether there are similar preferences over the images based on geographic proximity. We used the votes collected from 90,596 visitors from 147 countries. Each country is represented by a normalized vector that represents the percentage of votes their users expressed on Scary and Not Scary images.

We perform hierarchical clustering on the distributions of the countries. Figure 7 illustrates the dendrogram for top 30 countries based on the number of users. The cophenetic distance between each observation in the hierarchical clustering was found as 0.73. Some interesting cross-cultural trends can be observed from the clustering. For instance, neighbor countries United States and Canada are the most similar to each other in terms of preferences on the faces. A similar trend can be observed between neighbor countries Japan and Korea as well as Portugal and Spain. These observations indicate that the preferences over fear have a relationship with geographic proximity. Moreover, we used a heatmap-based approach to investigate the pairwise similarity between different countries. We used the top 10 images voted as Scary and represented each country as a 10-dimensional vector that includes the normalized voting information. Figure 8 shows the heatmap on top 30 countries on top 10 Scary images. We can observe that while most of the countries have a common consensus on the majority of the images, some countries found specific images highly non-scary such as Chile and China on Face#4. Another interesting observation is that geographically close-by countries China and Taiwan both found Face#3 and Face#4 as highly non-scary. These observations suggest that a cross-cultural trend on fear might exist, and different countries might have different opinions on what is Scary or Not.

Validation Study

While we received hundreds of thousands of indications that our machine-generated images were indeed scary, having subjects rate something as simply ‘scary’ or ‘not scary’ does not inform us of whether or not the images themselves actually induce the psychological construct of fear. Perhaps something that is casually rated as ‘scary’ actually alters mood, but it is also possible that by rating something ‘scary’ subjects are simply indicating that the image corresponds to the conception of what a frightful image typically looks like. Do the images we generated actually – psychologically – scare people? To investigate this question, we ran a validation experiment on Amazon’s Mechanical Turk that employed psychometrically validated measures of affect and anxiety.

We randomly assigned 752 subjects to three treatment arms. The first arm consisted of the ten images that received the most ‘scary’ votes (Scary). The second arm consisted of the ten images that received the fewest ‘scary’ votes...
Figure 8: Heatmap of countries and their probability of finding top 10 images as scary. One can observe that while most of the countries have a common consensus on the majority of the images, some countries found specific images highly Scary, such as Greece and Switzerland on Face#3 and New Zealand on Face#2.

Figure 9: Images used in experimental validation. The ‘Scary’ faces comprise the ten faces from the platform that received the most ‘scary’ votes. The ‘Less Scary’ faces comprise the ten faces that received the fewest ‘scary’ votes. The ten ‘Control’ faces were randomly drawn from the Chicago Face Database (Ma, Correll, and Wittenbrink 2015) set of neutral faces.

(Less Scary). For the third arm (Control), we randomly selected ten neutral expression faces from the Chicago Face Database (Ma, Correll, and Wittenbrink 2015) (see Figure 9). Because affect and anxiety vary differentially across gender (Thompson 2007), we block-randomized along with the gender of respondents for added statistical efficiency (Gerber and Green 2012).

Our validation study had two outcome measures. The first is a short form of the Positive and Negative Affect Schedule (I-PANAS-SF). The I-PANAS-SF is derived from the original twenty PANAS (Watson, Clark, and Tellegen 1988) item pool and allows us to measure – and separate – dimensions of positive and negative affect. It consists of ten items including five positive affective states: active, determined, attentive, inspired and alert and five negative affective states: afraid, nervous, upset, hostile and ashamed. Participants are asked to respond to the positive and negative states which describe their feelings. The second metric is a shortened version of the State-Trait Anxiety Inventory (STAI-SF) which measures subjects’ state anxiety. It is a psychological inventory based on a 4-point Likert scale ranging from not at all to very much and consists of six items assessing the degree that patients feel calm, tense, upset, relaxed, content and worried. The scores of all items are summed to produce a total score in which higher scores are positively correlated with greater anxiety. We randomized the order that our outcome measures were presented to the subjects. Finally, we pre-registered our experiment and analysis plan with AsPredicted.org as study #3410 and we follow that analysis plan below.

Negative/Positive Affect and State Anxiety

The results of our experiment indicate that our machine-generated faces produced substantial increases in negative affect and state anxiety as well as – to a lesser degree – worsened positive affect, as compared to our control condition. Respondents in the Scary and Less Scary conditions had markedly and significantly increased scores on the state anxiety measure (STAI-SF) as compared to Control, as can be seen in Figure 10 panels (a) and (b). Scary STAI-SF OLS coefficient is measured as 8.059 with a t-statistic: 6.346 and Cohen’s d: 0.58 while Less Scary STAI-SF OLS coefficient is measured as 6.336 with a t-statistic: 5.249 and Cohen’s d: 0.48. The Scary and Less Scary conditions did not significantly differ from one another.

The generated images also produced substantial and statistically significant increases in negative affect compared to the control faces (see Figure 10 panels (c) and (d)). Scary
PANAS negative affect OLS coefficient is measured as 3.513 with a t-statistic: 7.156 and Cohen’s d: 0.66 while Less Scary PANAS negative affect OLS coefficient is measured as 3.021 with a t-statistic: 6.412 and Cohen’s d: 0.59. The Scary and Less Scary conditions again did not significantly differ from one another on this measure.

Finally, the generated faces reduced positive affect as compared to the control group, with the Less Scary faces splitting the difference between the two other groups (see Figure 10 panels (e) and (f)) Scary PANAS positive affect OLS coefficient is measured as -0.898 with a t-statistic -2.03 and Cohen’s d: -0.21. The Less Scary condition did not significantly differ from the Scary or from the Control conditions.

**Heterogeneous Effects by Gender**

In addition to our main effects, we observe that female participants indicate significantly higher responses on the STAI scale in response to the Least Scary condition than do male participants. Note that our plan to investigate heterogeneous effects by gender was pre-registered in our AsPredicted.org plan #3410.

Male respondents in the Least Scary conditions had markedly and significantly reduced scores on the state anxiety measure (STAI-SF) as compared to female respondents in this condition, as can be seen in Figure 10 panels (g) and (h) (STAI-SF OLS male-by-least-scary interaction coefficient: -6.723, t-statistic: -2.666, Cohen’s d: -0.38). We observe no other significant heterogeneous effects by respondent gender.

Ultimately, the results of our validation experiment indicate that the generated faces – for both the Scary and Less Scary images – significantly and markedly increased psychometrically validated anxiety and negative affect as compared to the Control condition. Further, female respondents in our sample exhibit greater amounts of anxiety induced by the Less Scary condition.

**Conclusion**

As Artificial Intelligence makes strides in solving challenging analytical problems, many people believe that an important subset of human tasks such as inducing human emotions is still far from the ability of machines to outperform humans. In this work, we challenge this hypothesis and explored the potential of deep learning and crowd-sourcing to induce extreme emotions, in particular fear on a case study at nightmare.mit.edu. We create a high-volume dataset of emotional preferences using crowd-sourcing and use it to study fear in a variety of applications: we showed that we can build a model that learns which images are particularly Scary and distinguish between Scary and Not Scary images. We showed that while there seems to be a common consensus on the groups of Scary images, there exist several sub-groups among Non-Scary images. Moreover, we showed that latent preferences of the users towards Scary and Non-Scary images can be discovered using collaborative filtering approaches, which shows the potential to tailor personalized images that target specific users. We also explored cross-cultural preferences for fear to observe how preferences change based on geographical location. We ob-
served that some countries that are close to each other on a geographical level, such as America United States, and Canada, or Japan and Korea have the most similar preferences over the images. This observation suggests that there might be a cross-cultural competence over the images. In addition, while there is a global consensus on the majority of images, we observed that some countries found specific images highly Scary or highly Non-Scary which suggests that there might exist images that particularly affect certain cultures. Finally, we run a validation study where we performed a controlled experiment on \( n = 752 \) subjects on Amazon’s Mechanical Turk where we verify whether the generated images psychologically move people on psychometrically validated measures of effect and anxiety such as I-PANAS-SF and STAI-SF. Our exploratory results and validation experiment suggests that deep learning and generative algorithms have a significant potential for inducing emotions.

As future work, our approach can be extended to other types of emotions such as empathy. It can further be extended to improve the performance of the image generation system by tailoring the preferences towards particular users or can be explored to understand what particular features of the generated images induce certain emotions.

**References**


ISBN: 978-989-54160-3-5
Survival of the Wittiest: Evolving Satire with Language Models

Thomas Winters and Pieter Delobelle
Dept. of Computer Science; Leuven.AI
KU Leuven, Belgium
{firstname}.{lastname}@kuleuven.be

Abstract
Large pre-trained transformer-based language models have revolutionized the field of natural language processing in recent years. While BERT-like models perform exceptionally well for analytical tasks such as classification and regression, their text generation capabilities are usually limited to predicting tokens within a given context. In this paper, we introduce GALMET, a model that generates text by using genetic algorithms with BERT-like language models for evolving text. We use GALMET with the RoBERTa language model to automatically evolve real headlines into more satirical headlines. This is achieved by adapting the masked language head to the headlines domain for the mutation operator and finetuning a regression head to distinguish headlines from satire for the fitness function. We evaluated our system by comparing generated satirical headlines against human-edited headlines and just the fine-tuned masked language head. We found that while humans generally outperform the model, generations by GALMET are often preferred over human-edited headlines. However, we also found that only using the fine-tuned masked language model gives slightly preferred satire due to generating more readable sentences. GALMET is thus a first step towards a new way of creating text generators using masked language models by transforming text guided by scores from another language model.

In this research, we aim to investigate if we can improve the generative capabilities of BERT-like models by combining them with genetic algorithms. To achieve this, we introduce GALMET (Genetic Algorithm using Language Models for Evolving Text). To enable these textual transformations, this framework uses two different strengths of BERT-like models, namely their text classification capabilities and their masked language model. This study thus pilots a possible combination of BERT models’ masked language model head (which enables small textual modifications), with genetic algorithms’ mutation operators (which requires a function that slightly modifies an individual). It also simultaneously studies BERT models’ power for textual regression (thus labeling sentences with real-valued numbers) with genetic algorithms’ fitness functions (which require a function to label individuals, preferably with real-valued numbers). We then evaluate if the framework can be applied for evolving headlines into more satirical-sounding texts. While the results for humor were not incredibly satisfactory, the proposed mechanism could still enable a new way for creative language generation in several other domains as well (e.g. poetry or adversarial text generation).

Background

Language Models
BERT (Devlin et al. 2019) is a language model that uses the encoder stack of transformer models (Vaswani et al. 2017), which consists of multiple attention heads that correlate co-occurrences of words or tokens. For an analytical detail of the attention mechanism, see the introductory paper (Vaswani et al. 2017). It is highly suited for classification and regression tasks on an input sequence, ranging from named entity recognition to high-level sentiment analysis. The BERT model later got robustly evaluated and optimized in the RoBERTa model (Liu et al. 2019). These encoder-only language models are initially trained on the masked language modeling (MLM) task that is based on the Cloze task, where the training objective is to predict a masked word or token $T_i$ at a certain position based on the context. Interestingly, this can be interpreted as a probabilistic model, with the model generating a conditional distribution for each masked token $T_i$ following

$$Pr(T_i | T_1 \ldots T_{i-1}, T_{i+1} \ldots T_n).$$
These tokens are in most cases words, based on the frequency of each word appearing in a dataset. A common word can usually be expressed by a single token, while less frequent words are usually expressed as multiple tokens. The MLM task allows the model to learn linguistic knowledge in a self-supervised manner from unlabeled text sequences and is usually only used for pre-training. With transfer learning, a new head can be fine-tuned in a supervised manner to perform another type of language task.

Genetic Algorithms
The genetic algorithm is a prominent type of evolutionary algorithm that uses techniques inspired by natural selection to discover high-quality solutions for a search problem where solutions can be evaluated (Holland and others 1992). The algorithm generates an initial population of \( \mu \) individuals, evaluates its fitness and selects the best few for the next generation. These selected individuals are often crossed-over, where new individuals have elements from multiple parents, and are often slightly mutated. This continues until the stopping criteria are reached, such as the desired fitness value.

Satire Detection
Recently, several researchers released datasets for performing satire detection. Most notably, a dataset called “Humicroedit” contains headlines and the edits humans did to create funnier headlines, which were then rated by other humans (Hossain, Krumm, and Gamon 2019). In a competition, 48 different teams created models to achieve the best performance for estimating the perceived funniness of the edited headlines (Hossain et al. 2020a). As expected, most teams used pre-trained language models such as BERT and RoBERTa, with the winner using an ensemble of six different pre-trained language model architectures (Hossain et al. 2020a).

Satire Generation
There have been several research projects aiming to automate satire. One approach uses a genetic algorithm, that substitutes words from movie titles with words related to the satirical target to create satirical movie titles. An apprentice then learns to replicate from this algorithm and humans on Twitter creating the same type of movie title variations with this context using a neural sequence-to-sequence model (Alnajjar and Hamilalainen 2018). Other researchers also used the earlier mentioned Humicroedit dataset to train a transformer model to generate satirical edits to real headlines (Weller, Fulda, and Seppi 2020). Another recent approach used BERT summarization models that map true headlines, leading paragraphs and Wikipedia contexts to satirical headlines, achieving 9.4% funny headlines (Horvitz, Do, and Littman 2020).

Data
We combined datasets containing real and satirical headlines used previously in research and competitions, and added a funniness label. In this work, we use “real headlines” to refer to headlines that were created by actual news websites, and “satirical headlines” to refer to headlines from satirical websites. Some datasets also contain edited real headlines and edited satirical headlines, to be respectively more or less funny than the original. If a funniness rating was present in a dataset, it was normalized and used as a label, otherwise real headlines received label 0 and satirical headlines label 1. If a headline was already in the combined dataset, it was not included again. We first added datasets that contain rated funniness, namely 15K edited headlines from Humicroedit (Hossain, Krumm, and Gamon 2019), 8K from Funlines (Hossain et al. 2020b) and 2.7K from Unfun (West and Horvitz 2019). The first two edited real headlines by changing one word, while the third made satirical headlines less funny by changing as few words as possible. We also included unrated real and satirical headlines, namely 20K from Unfun, 26K from the Sarcasm Detection dataset (Misra and Arora 2019) and 22K from The Onion or Not dataset1.

After splitting into training, validation and test sets, the dataset was augmented by converting it to different casings (lowercase, uppercase, title case). Not only because this could give a hint to the original dataset (where for example lowercase was more prevalent), or because the edited headlines sometimes had a different case for the edit, but also because these sentences map to completely different tokens, and thus to different input sequences for a RoBERTa model. This resulted in a training dataset of 236k training, 30k validation and 34k test instances in their respective datasets. The distribution of the dataset is displayed in Figure 1, showing that it mostly contains real headlines and real satire.

RoBERTa Models
We fine-tuned two RoBERTa heads on the dataset, one regression model predicting the funniness rating and one masked language model on only the text sequences (referred to as Satire MLM in the remainder of the paper). The regression model achieves \( \text{MSE} = 0.0447 \) and \( R^2 = 0.548 \) on the held-out test set (see Figure 2 for the distribution). While the RMSE of 0.2113 seems surprisingly low compared to the Humicroedit competition winners (with RSME of 0.5016), this is due to our score normalization, and is also made incomparable due to our large dataset augmentation.

1https://www.kaggle.com/chrisfilo/onion-or-not
Table 1: The hyperparameter space used for finetuning all language models.

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>adam_epsilion</td>
<td>$10^{-8}$</td>
</tr>
<tr>
<td>adam_betas</td>
<td>$\beta_1 = 0.9, \beta_2 = 0.999$</td>
</tr>
<tr>
<td>fp16</td>
<td>False</td>
</tr>
<tr>
<td>gradient_accumulation_steps</td>
<td>$i \in {1, 2, 3, 4}$</td>
</tr>
<tr>
<td>learning_rate</td>
<td>$[10^{-6}, 10^{-4}]$</td>
</tr>
<tr>
<td>max_grad_norm</td>
<td>1.0</td>
</tr>
<tr>
<td>max_steps</td>
<td>-1</td>
</tr>
<tr>
<td>num_train_epochs</td>
<td>3</td>
</tr>
<tr>
<td>per_device_eval_batch_size</td>
<td>8</td>
</tr>
<tr>
<td>per_device_train_batch_size</td>
<td>8</td>
</tr>
<tr>
<td>seed</td>
<td>1</td>
</tr>
<tr>
<td>warmup_steps</td>
<td>100</td>
</tr>
<tr>
<td>weight_decay</td>
<td>[0, 0.1]</td>
</tr>
</tbody>
</table>

GALMET

We introduce the GALMET model, a Genetic Algorithm using Language Models for Evolving Text. We use the fine-tuned RoBERTa models to implement the genetic algorithm operators for a GALMET model that aims to transform headlines into more satirical counterparts.

Flow

GALMET starts by receiving an input sentence, here an initial headline. This sentence is tokenized and duplicated $\mu$ times as the initial population. GALMET then repeats several classic genetic algorithm steps, namely evaluating and selecting the best individuals, crossing and mutating them until a stop condition is achieved (Figure 3). In this case, evaluation happens by predicting the funniness of the sentences using the fine-tuned regression model as a fitness function, and calculating the Levenshtein distance function to prefer individuals with modifications to the right of that word in another individual. For the mutation operator, we created three different operators. The first is token substitution, which substitutes a token with a mask and uses the probability distribution from the Satire MLM to sample a replacement token. For example, for a sentence “The lion roars.”, the token for “lion” could be selected to be replaced by a mask to create “The <mask> roars.”, which is then filled in using the Satire MLM to create the sentence “The alarm roars.”. The second mutation operator is token addition, which randomly adds a mask in the sequence and fills it in similarly. The third removes a random token from the sequence. The algorithm stops when an individual receives a score from the regression model above a certain threshold, e.g. 0.99.

Table 2: The parameters used for transforming headlines into more satirical headlines using GALMET

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mu$ (population size)</td>
<td>50</td>
</tr>
<tr>
<td>$p_c$ (crossover probability)</td>
<td>0.2</td>
</tr>
<tr>
<td>$p_m$ (total mutation probability)</td>
<td>0.8</td>
</tr>
<tr>
<td>$p_m^{substitution}$</td>
<td>0.7</td>
</tr>
<tr>
<td>$p_m^{addition}$</td>
<td>0.05</td>
</tr>
<tr>
<td>$p_m^{removal}$</td>
<td>0.05</td>
</tr>
<tr>
<td>max # generations</td>
<td>30</td>
</tr>
<tr>
<td>goal fitness</td>
<td>0.99</td>
</tr>
<tr>
<td>max edit distance</td>
<td>7</td>
</tr>
<tr>
<td>max # elites</td>
<td>6</td>
</tr>
<tr>
<td>elite duplicates</td>
<td>3</td>
</tr>
</tbody>
</table>

An example execution of GALMET is shown in Table 3, where the sentence “Most Americans Want Congress To Investigate Michael Flynn” is transformed into “224 Americans Asked To Investigate Michael Jordan”. An example that ran for more iterations is given in Table 4, for which the evolution of the fitness functions is summarized in Figure 4. As expected, the edit distance rises, but is limited thanks to the Pareto front. We can see that the desired regression score is achieved at generation 14, but that the edit distance was 8, and thus not sufficient for the stop condition demanding an edit distance of 7 or less. We can also see that the regression score stays near 0, and then quickly jumps to the 1 regions, a clear bias originating from the binary datasets with little values between 0 and 1 (as illustrated in Figure 2).
Table 3: Example of a Pareto front after 5 iterations, containing the highest scoring mutation for each edit distance to the original. Here, the sentence from \(d_{edit} = 0\) is thus transformed into the sentence for \(d_{edit} = 4\).

Table 4: An example that ran for 16 iterations (see Figure 4).

**Evaluation**

We evaluated GALMET by transforming specific headlines into more satirical versions and checking which ones are funnier than human-created satirical versions. We also evaluated if the genetic algorithm is better than its mutation operator component by making it compete with our Satire MLM. Given a headline, the MLM head applies the substitution mutation operator an equal number of times as \(G_{ALMET}\) operator component by making it compete with our Satire MLM.

**Future Work**

While GALMET did not display outstanding results in this particular study, we still believe it can be the basis for a powerful mechanism for transforming textual sequences. There are several improvements for the satirical headlines domain we envision. First, it would be beneficial to improve the dataset balance by removing some binary class datasets, as they bias the regression model strongly towards either close to 0 or to 1. Second, only allowing real words to appear in sentences, by only replacing full words, and allowing to insert a random number of neighboring masks instead of single masks when using the mutation operators. Third, adding a detector to steer away from broken sentences, which is achievable for a RoBERTa-based model in a humor detection setting (Winters and Delobelle 2020). Fourth, it would

---

**Code**

The code, fine-tuned RoBERTa models and further implementation, training and parameter details are available on https://github.com/twinters/galmet.
be beneficial to either improve the cross-over operator to create better individuals than our current cross-over operator or just leave the cross-over out and use the components in a search setting instead of genetic algorithms.

We expect the GAML framework to be interesting for other text generation domains too, such as poetry generation or generating adversarial examples to textual classifiers. The framework could also have a use in co-creative applications by suggesting improvements to given text sequences.

Conclusion

We investigated how to improve the generative capabilities of analytical language models by combining them with genetic algorithms. For this, we created a novel text generation method for evolving text into text from a different domain, in our case, transforming headlines into satirical headlines. To achieve this, we introduced several new genetic operators based on pre-trained language models. On evaluation, we found that it performs similar to one of its components, and identified several causes and potential solutions. We believe this framework could open the way for novel co-creative applications where users can evolve their texts towards particular goal text domains, even if that domain might usually be hard for computers to grasp, such as poetry or humor.

Acknowledgments

We would like to thank the volunteers for judging the headlines in the human evaluation. Thomas Winters is a fellow of the Research Foundation-Flanders (FWO-Vlaanderen, 11C7720N). Pieter Delobelle was supported by the Research Foundation - Flanders (FWO-Vlaanderen) and also received funding from the Flemish Government under the “Onderzoeksprogramma Artificiële Intelligente (AI) Vlaanderen” programme.

References


Evaluating Natural Language Descriptions Generated in a Workspace-Based Architecture

George A. Wright
1Cognitive Science Research Group
School of Electronic Engineering and Computer Science
Queen Mary University of London
george.a.wright@qmul.ac.uk

Matthew Purver
2Department of Knowledge Technologies
Jožef Stefan Institute
Ljubljana, Slovenia
m.purver@qmul.ac.uk

Abstract

This paper concerns the evaluation of a workspace architecture for generating natural language descriptions, including methods for evaluating both its output and its own self-evaluation. Herein are details of preliminary results from evaluation of an early iteration of the architecture operating in the domain of weather. The domain is not typically seen as creative, but provides a simple testbed for the architecture and evaluation methodology. The program does not yet match humans in terms of fluency of language, factual correctness, and how completely the input is described, but human judges did find the program’s output easier to read than human generated texts. Planned improvements to the program also described in the paper will incorporate self-monitoring and better self-evaluation with the aim of producing descriptions that are more fluently written and more accurate.

Introduction

This paper describes work towards a self-evaluating architecture for language generation first described in (Wright and Purver 2020) and a method for evaluating the architecture by comparing human judgements of its output with its own self-evaluation. This iteration of the architecture operates in a toy domain: making simple descriptions of temperatures on a static, two-dimensional map but serves as an initial framework on which future versions performing more ambitious tasks can be built.

Theoretical Background

According to Fauconnier (1994), linguistic meaning is organized in mental spaces and according to Fauconnier and Turner (2002), creativity involves the projection of structures across mental spaces, often with the help of frames. Such processes cannot involve a deterministic search for an optimum, but instead a constant competition between structures evolving in a bubble chamber of mental spaces, only some of which become available to consciousness (Fauconnier and Turner 2002, p.321). The architecture described below implements the projection of structures across spaces while making use of an enzymes-in-cytoplasm metaphor of cognition similar to that proposed by Barrett (2005) which allows for a chaotic interaction of processes in a shared workspace or bubble chamber. These include processes of language production and comprehension which also interact when humans use language (Pickering and Garrod 2013). In this architecture, self-comprehension and self-evaluation are important because they help to determine which of the competing intermediate structures are used in future processing. An overall satisfaction score also affects how randomly processes occur. Evaluation of this architecture therefore takes into account not only the finished outputs of the program, but also its method for self-evaluation.

The Planned Architecture

The architecture has a bubble chamber and a coderack. The bubble chamber contains a network of concepts, frames, and their instantiations spread across a number of conceptual and working spaces. These are the long- and short-term memory of the program. The best, most useful structures bubble to the top of the program’s attention as their activation increases. The coderack, borrowed from Copycat (1993) and related work (Hofstadter and FARG 1995) contains a collection of codelets, (small tasks to be carried out), each of which has an urgency influencing the likelihood it runs. Codelets correspond to the enzymes of Barrett’s metaphor. They are selected from the coderack with a degree of randomness determined by the program’s satisfaction, a score of the quality of active structures in the bubble chamber (a structure’s quality is determined by evaluation codelets). High satisfaction leads to less random codelet selection thus more deterministic processing whereas low satisfaction leads to more randomness and opens a broader set of pathways to be explored. Self-evaluation is central to the architecture and is therefore important to consider when judging its performance.

Most codelets make a small change to the bubble chamber, for example by building a new node or link, or by changing a structure’s activation. All structures, including representations of the input, parse trees, and output text are built incrementally in this manner. Codelets also change the coderack by adding a follow-up codelet. Some codelets operate exclusively on the coderack by adding or removing codelets in order to ensure that the coderack does not become empty or overcrowded.

This style of architecture shares similarities with models
based on Baars’ (1997) Global Workspace Theory such as (Misztal and Indurkhya 2014) which has experts performing tasks in a shared workspace. But, where as codelets in this architecture are restricted to performing small operations, some experts in Misztal and Indurkhya’s architecture such as the metaphor expert operate at a much higher level and perform tasks comparable in complexity to work performed by a large collective of codelets.

**Engagement and Reflection Cycles**

According to Sharples (1998), the creative writing process involves a cyclic alternation between engagement (producing new ideas) and reflection (evaluating work so far). This has been implemented in models of language generation (Pérez y Pérez and Sharples 2001) as well as other models of creativity (Pérez y Pérez, de Cossío, and Guerrero 2013). The E-R model is a relatively high-level view of cognition which does not recognize the more intertwined nature of production and comprehension described by Pickering and Garrod (2013).

This architecture contains something like an engagement-reflection cycle but at multiple levels of abstraction and, due to the stochasticity of the coderack, with less rigidity.

**Codelet Cycles**

Most codelets operating in the bubble chamber belong to one of four types: suggesters, builders, evaluators, and selectors.

Suggesters find an element in the input such as a temperature on a map and suggest a possible structure that can be built for that element. For example, a temperature could be labeled as HOT or in the SOUTH, two temperatures could be combined into a single chunk if they are similar, two temperatures could be connected with a MORE or LESS relation, or a SAMENESS correspondence could be recognized between a chunk in the input and an item in a frame.

Having performed a classification, a suggester codelet places a builder codelet on the coderack with an urgency matching its confidence in its suggestion. If the builder codelet is run, the relevant structure is built and the builder codelet then places an evaluator codelet on the coderack.

Evaluator codelets determine the quality of the structure according to the same classifier as the suggester. Since certain classifications can be context dependent, for example a part-of-speech label may depend on how a word is used in a sentence, the classification of a structure by the time the evaluator is run may differ from when the structure was first suggested. The evaluator assigns a quality score to the structure and then places a selector codelet on the coderack.

Selector codelets compare two competing structures, for example two incompatible labels, and boost the activation of one while depressing the activation of the other such that only one structure is likely to be used in further processing. Higher quality structures are more likely to receive a boost in activation. Selector codelets also place another suggester on the coderack thus completing a cycle at the fine-grained level of workspace structures.

If a codelet fizzes because the bubble chamber does not contain the right conditions or if a follow-up has low urgency and never runs, the cycle breaks. Meanwhile new cycles are created as factory codelets add new suggesters and evaluators to the coderack so that processing does not stop prematurely.

![Figure 1: The lowest-level “cycle of engagement and reflection” at the level of individual nodes and links in the bubble chamber.](image1)

**View Cycles**

The architecture implements the simplex networks of Fauconnier and Turner (2002, p.120-2), which connect elements in an input space to elements in a frame and then elements in both the input and the frame to new elements in an output space. Since this is a language generating program, the frames are templates with slots to be filled in according to the input. The output is a text which describes the original input using the template structure. Each network exists within a view based upon the Worldview of the Tabletop model of analogy-making (French 1995). All structures within a view must be consistent with one another.

The architecture also uses views for self-monitoring. Monitoring views contain an output text, a semantic parse of the text and a set of correspondences between elements of the parse and the original input. The purpose of a monitoring view is to check that a text both makes sense and is an accurate description of at least part of the original input.

Texts which have been matched to part of the original input are made available for further processing inside higher level simplex networks using discourse frames. This allows for a recursion of simplex networks as described by Fauconnier and Turner (2002, p.151) and produces a cycle of engagement and reflection at the higher level of fragments of text which emerges from the cycles of engagement and reflection at the lower level of individual nodes and links.

![Figure 2: A higher level “cycle of engagement and reflection” at the level of pieces of text.](image2)

**The Current Implementation**

The implementation of the architecture described above operates within a simple domain, describing temperatures on a map. This requires a small knowledge base and allows for focus to be placed on the mechanisms of the architecture.
Implementation is still in an early stage and lacks much of
the self-monitoring provided by monitoring views.

In order to get the program to output text, a publisher
codelet occasionally runs, which finds templates that have
had their slots filled and outputs the resulting text. Current
outputs are therefore short and lack discourse structure, but
the evaluation of these outputs provides a base-line upon
which future iterations of the model can improve.

The current implementation’s satisfaction is calculated as
the mean of the product of each bubble chamber structure’s
quality and activation. This means the satisfaction is higher
when the most active structures have a high quality and lower
when active structures have a low quality or high quality
structures have a low activation. But, as discussed below, this
results in a satisfaction score which fails to take into account
a more global perspective on the bubble chamber.

**Evaluating The Program**

The relatively transparent nature of the program allows it to
be evaluated in a number of ways: the intermediate representa-
tions it builds when processing the input, its textual output,
its understanding of its own textual output (through syntactic
and semantic parses), and its satisfaction score for its output
can all be seen and evaluated by external observers.

Below is described a subjective and intrinsic evaluation
of outputs of the system implemented thus far - a survey
which evaluated the system in isolation from any practical
application and according to human value judgements. Such
surveys commonly focus on two main criteria: the quality
of a text, and its accuracy relative to the input (Gatt and

**The Survey**

Human subjects in the survey were asked to compare two of
the program’s outputs for each input. They had to answer
four questions for each pair:

1. Which text is easier to understand?
2. Which text is more fluent?
3. Which text is more factually correct?
4. Which text represents the map more completely?

Respondents could answer each question in one of three
ways: the first text is better than the second, the second text is
better than the first, or the two texts are approximately equal.

The aim of the first two questions was to capture the lingui-
sitic quality of the texts, while the aim of the final two
questions was to capture their accuracy as descriptions of the
input map. Survey respondents only saw the map after
the first two questions so that any inaccuracies in the description
would not influence the quality score.

Human subjects had to compare two outputs rather than
score them on a scale as it is unclear what the criteria are
for high or low scores, especially when viewing the first few
outputs from a program. Furthermore, Belz and Kow (2010)
compared preference-based evaluation to score-based evalu-
ation and found that preference-based evaluation results in
less variance between respondents.

Since the computer program provides its own satisfaction
score for its work, human evaluation can also be used to
check if its internal measure of satisfaction matches with
human judgements or if its method for calculating satisfaction
could be improved. Since the program only has a single
number to describe its “satisfaction”, there is no one-to-one
 correspondence with the questions used to judge linguistic
quality and factual accuracy. The score is also an absolute
number rather than a preference judgement. Nevertheless,
rankings based on human judgements and rankings based on
the program’s internal score ought roughly to align.

Methods for evaluating the creativity of computer pro-
grams commonly try to rate the novelty of outputs as well
as their quality, see for example (Ritchie 2007). This is not
attempted here since the domain is so simple and the outputs
are so short that no output is likely to be in any way novel.
It is hopefully clear though, that this architecture could in
theory be applied to a more complex domain that would al-
low for more exciting outputs where novelty would be worth
considering.

**Generation of Texts for the Survey**

![Figure 3: The first input as displayed to survey respondents. Numbers show temperatures in centigrade.](image)

The survey was carried out using four different inputs to
the program. For each input, the program was run ten times,
and three outputs were randomly selected. Outputs all took
the form of simple statements of fact. Added to these outputs
were two human-generated descriptions which were gathered
from a separate survey. For each input, one description was
selected which was written with detailed, full sentences while
the second description was brief and often written in note-
form. At no point were the respondents told that they were
evaluating machine-generated or human-generated text. The
texts used for the first input were:

- **A** (Human) “The temperature is cold in the north but progres-
sively warm moving south, reaching 24 degrees.”
- **B** (Computer) “It is hot in the southeast.”
- **C** (Computer) “It is mild in the northeast.”
- **D** (Computer) “The north is mild.”
- **E** (Human) “Cool in the north, warm in the south.”

The purpose of including human-generated outputs was
to check that respondents (on Amazon Mechanical Turk)
understood the task and were not pressing random buttons. A respondent who understands and pays attention to the task ought at this point broadly to prefer the human-generated texts. In future, improved iterations of the program ought to surpass the briefer note-like human-generated texts. Outputs of future iterations can also be compared to outputs of the current iteration to check if changes to the architecture result in improved results.

### Results of the Survey

The results of the survey are unsurprising in that they show that the program is overall below human-level performance, but they also highlight certain issues that should be taken into account in future evaluation.

It should first be noted that respondents of the survey did not show a high degree of agreement. The Fleiss’ Kappa scores were 0.342 for ease of understanding, 0.238 for fluency, 0.484 for factual correctness, and 0.485 for completeness (to calculate Fleiss’ Kappa the three possible answers to each question were treated as a category). This may in part be due to the fact that respondents had a different understanding of the questions they were being asked: future surveys should make more clear what each of these terms means, especially correctness and completeness which some respondents seemed to treat as the same. Low agreement may also have been caused by arbitrary decisions being made when similar computer outputs were compared. The survey also only had 7 respondents. In future, surveys using more respondents may result in better agreement.

Respondents on average, ranked human-generated texts above computer-generated texts along the dimensions of fluency, correctness, and completeness. But they found computer-generated texts easier to understand. A similar result was found by Reiter et al. (2005, p.138) who found that readers preferred a computer program’s weather forecasts to those written by human’s due to greater consistency in the program’s word choices. It is likely to be the case that more rigid and precise computer programs will always outperform humans along this dimension within small data-to-text applications, but this should be less easy to achieve in more complex domains requiring narrative or explanation. Achieving greater ease-of-understanding scores will therefore not be a priority in future work on this architecture where the aim is to achieve something closer to human-like creativity in language generation.

For the most part, no preference was shown for one text’s easiness or fluency over another when two computer-generated outputs were displayed side-by-side. This is understandable given that computer-generated outputs all followed one of two sentence patterns: the [location] is [temperature] and it is [temperature] in the [location]. Some computer-generated texts used words which did not match well with the input map and were therefore not preferred when it came to correctness and completeness.

There may have been some confounding variables which affected respondents’ evaluation of the text, for example the length of the sentences being compared. Future evaluation should consider the extent to which such variables influence people’s preferences.

### Evaluating the Program’s Self-Evaluation

The linguistic similarity of the outputs is reflected in the computer program’s satisfaction scores. The 40 runs executed for the purpose of evaluation had a mean satisfaction score of 0.704 with a standard deviation of 0.065. But, the program even had similar satisfaction scores in the 12 cases when it failed to produce an output before timing out after 30,000 codelets were run. This is because the satisfaction score is based entirely on the quality and activation of individual, low-level structures in the bubble chamber and does not take into account more global criteria for satisfaction such as the proportion of the input that has been described. It is clear that an improved metric for the satisfaction of the program is required but unfortunately it is difficult to compare different metrics when the program consistently produces similar outputs.

### Future Work

There are many improvements that can be made to the architecture, most urgent of which is the implementation of monitoring views in which codelets will build correspondences between the semantic parse of a text and the original input in order to check whether or not the text is factually correct and also to measure the extent to which the input has been described. This should reduce the incidence of inaccurate outputs.

The addition of discourse frames which the program can use to combine phrases and produce longer sentences should result in more fluent and complete descriptions of the input.

Furthermore, changes in higher level structures such as greater coverage of the input and improved discourse structure must be reflected in the program’s satisfaction score. Future rounds of evaluation can consider alternative methods for calculating satisfaction and compare human rankings with the program’s scoring of its own output.

### Conclusion

This paper has provided the outline of a planned architecture for language generation and a method for evaluating the architecture by eliciting human judgements of its output and comparing those judgements to the program’s internal self-evaluation. Described in the paper is an early iteration of the architecture which lacks some of the core components required for self-monitoring and more complex discourse.
structuring. The program’s outputs are therefore still disappointing, but outputs of future versions of the program can be compared with its current outputs to see the extent to which greater self-monitoring improves performance.

Acknowledgments

Purver is partially supported by the EPSRC under grant EP/S033564/1, and by the European Union’s Horizon 2020 program under grant agreement 825153 (EMBEDDIA, Cross-Lingual Embeddings for Less-Represented Languages in European News Media). The results of this publication reflect only the authors’ views and the Commission is not responsible for any use that may be made of the information it contains.

References


Assessing MultiPlot Stories: from Formative Analysis to Computational Metrics

Pablo Gervás, Eugenio Concepción, Gonzalo Méndez
Facultad de Informática
Universidad Complutense de Madrid
Madrid, 28040 Spain
{pgervas,econcepc,gmendez}@ucm.es

Abstract
Recent interest in story generators capable of combining more than one plot line into an elaborate story have been handicapped by the lack of either theoretical material or quantitative metrics to ascertain the quality of outputs of such attempts. The present short paper postulates a set of metrics designed to capture some of the insights elaborated during a formative evaluation of an existing attempt at plot weaving.

Introduction
The mechanics of how to combine more than one plot line into a rich story have become a subject of interest in storytelling research in recent times. Solutions have been proposed to address the task (Fay 2014; Porteous, Charles, and Cavazza 2016; Gervás 2014a; 2018). Yet there is a shortage of either theoretical material or quantitative metrics to ascertain the quality of outputs of such attempts. The present short paper postulates a set of metrics designed to capture some of the insights elaborated during a formative evaluation of an existing attempt at plot weaving. The metrics are calibrated against qualitative evaluations by human judges and tested over outcomes of baseline solutions for plotline weaving. The metrics emulate the observations made by human judges in that they consider separate sets of positive and negative features. The metrics are designed to identify features that at some point in the formative evaluation have been deemed by some human judge to either add or detract to the perceived value of a story. The overall judgment on a given story must be extrapolated from the corresponding collection of features.

Related Work
The three topics considered relevant for this short paper are prior solutions for plot line combination, quantitative metrics for stories and formative evaluation of plot weaving.

Plot Line Combination
The systems reviewed here all combine a number of “plot lines” in some form, but each uses a different terminology for referring to them. To facilitate description, we consider an abstract concept of plot line as a sequence of plot elements, each describing an event relevant to the structure of the story, with the possible addition of a set of roles that characters play in the event.

Fay (2014) considers a plot weaving algorithm that builds new stories combining plot lines for a set of given character types. The system finds the character models best matching the given types, retrieves narrative threads associated in the corpus with those models, and finds the best combination of those narrative threads into a single story, ensuring that characters’ plots are compatible and that the resulting timeline is consistent.

Porteous, Charles, and Cavazza (2016) presents an interactive storytelling system that constructs stories with multiple interleaved plot lines. Their system constructs the stories dynamically using a plan-based approach in response to set of input parameters that drive the number of plot lines to be interleaved and the relative time spent on presentation of each subplot.

The StoryFire (Gervás 2018) system generates stories inspired by the movements of pieces in a chess game. This system combines concepts of narrative thread – sequence of predicates affecting a given piece – and plot line – a linear sequence of abstract labels for plot-relevant events that may describe an interesting story line. In this case, the plot line (usually single) is used to inform an interweaving of narrative threads for different characters.

Computational Metrics for Stories
Existing previous work on quantitative metrics for stories has not addressed multi-plotline stories explicitly. The work in (Gervás 2014b) describes a number of metrics to quantify a set of desired structural features over narrative renderings of game logs, and it focuses on issues such as coverage of the game, and features like redundancy and continuity of the composed discourse. Earlier works focused on metrics for story novelty (Peinado et al. 2010) and related concepts such as similarity between stories (Hervás et al. 2015). In particular, (Hervás et al. 2015) describes a calibration process based on comparing results on the metric against human judgement.

Formative Evaluation of Plot Weaving
The work of (Concepción, Gervás, and Méndez 2020) explores baseline solutions for weaving together a set of plot
templates into stories where scenes from the different templates appear interleaved. A plot template would correspond to the plot line we are considering — these plot templates include additional information on roles played by the characters. Several procedures for combining plot templates are described, some based on existing literary techniques (Communicating Vessels, Chinese Boxes) and some presented as baselines for computational approaches to the task (subplot concatenation, subplot alternation, and random mixing). A formative qualitative evaluation of 10 story examples is included. This evaluation includes qualitative analyses by human judges of the stories in question, where specific features that add or detract to the perceived value of the story are discussed.

Automated Emulation of Human Assessment of Plotline Weaving

The present short paper describes a set of metrics designed to capture in a numerical form the insights that arose from the formative evaluation presented in (Concepción, Gervás, and Méndez 2020). This formative evaluation uncovered insights at two different levels: features perceivable in stories that are considered valuable by human evaluators, and types of knowledge about the story that are being brought into play by human evaluators when making such judgments.

Insights on Desirable Features in Multi-Plotline Stories

The insights that have been considered relevant for the quality of plot line weaving, and susceptible of numerical formulation are described below.

The comments in the formative evaluation made it clear that there are two features of the stories that play an important role in the perception that human judges have of the quality of their weaving: the valence of characters (whether they are good or evil) and the level of activity conveyed by each scene.

Evaluators praised stories where sub-plots have been combined merging villain with villain or hero with hero.

They also praised stories where descriptive scenes from one plot line were interleaved with descriptive scenes from another, and active scenes were interleaved together. This intuitively leads to a story that switches from a more descriptive mode to a more narrative mode at one point, and the subplots that make it up align in that sense.

Another feature that was considered relevant is the rhythm of alternation between sub-plots when they are interwoven. Evaluators praised stories in which the rhythm of alternation between subplots — how many scenes from each subplot are told together before switching to the other — matches the perceived impression of activity for the story. If scenes are active, and significant events are happening in each sub-plot, switching between sub-plots can happen every few scenes; whereas if scenes are descriptive and nothing much actually happens in each one, more time should be spent on each sub-plot before switching to another.

Two further features were mentioned as positive for some stories: the existence of an overarching plot for the story that starts and ends the story, and the appearance of a complete sub-plot as an insertion within another.

It is important to note that, when asked to assess stories, human judges did not resort to scoring them or ranking them, rather made a set of observations on each story. These observations were either positive (identifying positive features in the story) or negative (identifying negative features in the story). The metrics that we are proposing follow this same pattern.

Knowledge about Stories Relevant to Multi-Plotline Assessment

The analysis of the formative evaluation suggested that valence of the characters and level of activity of scenes are relevant features that need to be made available to a system hoping to assess multi-plotline stories. Therefore the existing set of resources was hand annotated with values for these features. A baseline annotation was carried out over the templates for sub-plots as a first approximation. In this way, the relevant information is tied in to each plot template.

Valence for characters in a given scene was annotated with a value of -1 for characters performing evil actions and 1 for characters performing good actions. A valence value of 0 is assigned by default to all other characters.

Level of activity of scenes was annotated by adding a flag to scenes in a template that involved some relevant action. The rest of the scenes are considered descriptive.

Quantitative Metrics for Multi-Plotline Weaving

The system as it stands can parse stories written from text files in a particular format into a representation in terms of templates built of scenes. It also allows construction of new stories by combining a number of plot lines using the baseline computational strategies described in (Concepción, Gervás, and Méndez 2020). In both cases the representation that is obtained allows for the automated compilation of numerical data for character valence and activity based on the annotations described.

The procedure constructs four different types of vectors of numerical values for each story:

- **vectors of character valences**: for each character, compile the sequence of valence values for the scenes in the story
- **vectors of sub-plot alternation**: for each span of the story corresponding to a different sub-plot, note which template it comes from
- **vectors of alternation rhythm**: for each span of the story corresponding to a different sub-plot, note its length in number of scenes
- **vectors of matching scene activity**: for each of the spans in the alternation rhythm sequence, compile the count of active scenes

Over these vectors, a number of features considered by the human evaluators can be computed automatically. In all cases, the philosophy is to identify features that at some point in the formative evaluation have been deemed by some human judge to either add or detract to the perceived value of a story.
Table 1: Story 3 combines a Creation of Life (CL) subplot with a Destructive Outsider (DO) subplot. Scene labels from each subplot are shown in [square brackets], in bold if negative in valence. Active events underlined.


Table 2: Story 9 combines a Creation of Life (CL) subplot with a Destructive Outsider (DO) subplot.

The automatic identification of the following features has been implemented:

- overarching plot (vector of sub-plot alternation starts and ends with the same sub-plot)
- inserted sub-plot (sub-plot appears only once in vector of sub-plot alternation)
- spans with regular interweaving rhythm (a given value of alternation rhythm is maintained over a number of transitions between sub-plots)
- rhythm matched to activity (either slow rhythm for spans with low activity, or high rhythm for spans with high activity)

In addition, the values for valence of characters are used to build an overall pattern of alternation between valences is built for a story. This allows the establishment of distinctions between stories that end events with negative valence (tragedies) and stories that end in events with positive valence (comedies, rags to riches stories, overcoming the monster stories . . .).

Discussion

The proposed metrics are calibrated against the inspiring stories and tested over automatically generated stories.

Calibration over Inspiring Stories

The results for the proposed metrics over the inspiring stories considered in the formative evaluation of (Concepción, Gervás, and Méndez 2020) are presented in Table 3.

The application of the metrics to these stories is intended as a calibration exercise, to test whether the metrics indeed capture the intuitions that inspired them. Observations on story quality are not considered because the formative evaluation used as reference did not explicitly consider them.

The set of stories includes examples of accepted strategies used in literary text (Chinese Boxes inserts a complete sub-plot as a single span within another. Communicating Vessels interleaves several subplots with different rhythms). These strategies represent instances of complex weaving strategies that are considered valuable. The metrics clearly identify the Chinese Boxes strategy in stories 6, 7, 9 and 10 (by design the results include both overarching plot and inserted plots).

Story 9 has a span of identified rhythm (rhytSp = 1) has a similar situation towards its end (two contiguous spans of 4 scenes) and these also happen to include no activity so they are recognised as a slow pace segment (slow = 1) of the story, with relatively slow sub-plot alternation matching scenes low in activity. Story 9 has a similar situation with spans of 3 scenes, but the activity in that case is not regular. Examples of these stories are shown in Tables 1 (Story 3) and 2 (Story 9). The examples have been chosen using the same subplots to allow comparison of the differences in structure between the resulting complete story.

The Communicating Vessels strategy exercises greater freedom in the way it combines subplots, allowing it to choose whether to include an overarching plot (story 2) or not (stories 5 and 8). Because it interweaves subplots more freely, it can result in a higher number of regular rhythm spans (rhytSp, see story 8).

The Alternation strategy by design imposes a fixed rhythm of alternation (rhytSp) leading to a single span of regular rhythm of the same size as the story (spSp, see Story 1).

The Random strategy has the potential to replicate the freedom of the Communicating Vessels strategy, as shown by the similar values shown by the metrics for rhytSp and spSp.

The patterns for valences show a marked tendency towards positive endings (7/10) over negative ones. This is a natural consequence of the nature of the templates considered (only 1/4 ends on a negative valence). Overall there is a marked tendency to start stories on a negative note (the classical solution of starting with a conflict to be resolved).

Testing over Generated Stories

The results of testing the proposed metrics over a larger set of automatically generated stories are shown in Table 4.
The stories are generated using the three baseline computational strategies described in (Concepción, Gervás, and Méndez 2020): concatenation, alternation and random. Results are reported as totals over a set of 20 generated stories for each strategy.

The values for the metrics serve to highlight the shortcomings inherent in the baseline weaving strategies. The Concatenation strategy allows neither overarching plots nor inserted plots. Regularities in rhythm arise by serendipity whenever (at least two of) the subplots involved have the same length. Because the spans involved are always long, the pace of switching between subplots is identified as slow. The Alternation strategy allows overarching plots in certain cases (50% of the time when two plots are used, namely when one of the is longer than the other) and does not allow inserted plots. Due to its nature, it generates a single span with a regular rhythm of alternation of the same size as the story – which varies depending on the templates employed. The templates available do not include continuous sequences of active scenes, so identifying patterns of activity at that rhythm is almost impossible. The Random strategy does allow both overarching plots and inserted plots, and it allows the appearance of spans of regular rhythm in different patterns. The results of this strategy sometimes include several spans of interweaving at different rhythms. In this sense, it is the only of the computational strategies tested that can emulate the behaviour of the Communicating Vessels reference strategy. The Random strategy does have shortcomings of its own in that it is altogether blind to any features that it might be introducing.

With respect to valences, the reported outputs are built using a larger set of templates, with higher prevalence of evil acts towards the end (in 4 out of the 7 templates used). This leads to a higher average of evil ends (slightly over 4 out of 10 as opposed to the 3 in 10 of the hand-crafted stories). The set of templates used is chosen at random, which leads to a more even distribution (5 out of 10 in average) of positive vs. negative beginnings. The values for the hand-crafted stories may have been affected by the original decision to rely on a restricted set of similar templates throughout to make it easier to perceive changes in structure resulting from different weaving strategies.

An example of a generated story is shown in Table 5. This story presents a number of the features that are identified by the metrics proposed in this paper. There is a regular rhythm span with three scenes from the Split Personality Comic subplot (SP1 to SP3) followed by three from the Creation of Life (CL3 to CL5) subplot. There is another regular rhythm span with two scenes from the Split Personality Comic subplot (SP4 and SP5) followed by three from the Creation of Life (CL6 and CL7) subplot. There is no overarching plot, because the story starts with a scene from the Creation of Life subplot and ends with a scene from the Split Personality.

Table 5: Generated Story 2 for Random combination of 2 subplots: combines a Creation of Life subplot and a Split Personality Comic subplot.

Comic subplot. This, together with the choice of subplots, implies that the story has a positive start and a positive end.

The story also includes a number of additional features that are not covered by the metrics but which are clearly relevant for the task. The formative analysis of the stories in (Concepción, Gervás, and Méndez 2020) identified the problem of inconsistency between the life spans of characters unified between two subplots – characters that die in the final story as required by one of the subplots but then continue active as required by another. The story in Table 5 presents two instances of similar phenomena: (1) Edward falls in love with Martha (SP3) and then Edward falls in love with Scott (CL4), and (2) Edward kidnapped (CL5) but remains active without having been released (CL4 and SP3). This type of issue needs to be addressed in further work. It will very likely require further enrichment of the resources with information on when characters are restricted in movement or fall in love.

It is also interesting to note, that, given the peculiarity of Split Personality Comic subplot the characters Edward and Hans are both the same person and separate characters. This complicates computation of this type of consistency restrictions.

Intended Application of the Metrics

The metrics reported here are proposed as a first step towards devising a set of informed weaving strategies that aim to produce stories that exhibit the features identified as desirable. This short paper reports on the enrichment of the underlying resources and the development of the metrics and constitutes a preliminary result. Further work will explore weaving strategies that may take advantage of both of the contributions reported here (enriched knowledge resources and computational metrics for desired features) to achieve multi-plotline stories exhibiting the features deemed valuable by the human judges during the formative evaluation used as inspiration.

Solutions for the automated extraction of the knowledge resources from corpora of stories will also be explored.

Conclusions

This short paper reports these preliminary results on the metrics. The proposed metrics built automatically do serve to identify the features in the inspiring set of stories that they were intended to capture. The baseline solutions for plot weaving considered produce unimpressive output scored low by the metrics. Small peaks in the score do seem to match serendipitous good features observable in the output stories.

Ongoing efforts exist to develop plot weaving solutions to optimise these metrics. It is hoped that such plot weaving solutions will lead to significant improvements in the outcomes.

Acknowledgments

This paper has been partially funded by the projects CANTOR: Automated Composition of Personal Narratives as an aid for Occupational Therapy based on Reminiscence, Grant. No. PID2019-108927RB-I00 (Spanish Ministry of Science and Innovation) and InVITAR-IA: Infraestructuras para la Visibilización, Integración y Transferencia de Aplicaciones y Resultados de Inteligencia Artificial, UCM Grant. No. FEI-EU-17-23.

References


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.


Collaborative Storytelling with Human Actors and AI Narrators

Abstract

Large language models can be used for collaborative storytelling. In this work we report on using GPT-3 (Brown et al. 2020) to co-narrate stories. The AI system must track plot progression and character arcs while the human actors perform scenes. This event report details how a novel conversational agent was employed as creative partner with a team of professional improvisers to explore long-form spontaneous story narration in front of a live public audience. We introduced novel constraints on our language model to produce longer narrative text and tested the model in rehearsals with a team of professional improvisers. We then field tested the model with two live performances for public audiences as part of a live theatre festival in Europe. We surveyed audience members after each performance as well as performers to evaluate how well the AI performed in its role as narrator. Audiences and performers responded positively to AI narration and indicated preference for AI narration over AI characters within a scene. Performers also responded positively to AI narration and expressed enthusiasm for the creative and meaningful novel narrative directions introduced to the scenes. Our findings support improvisational theatre as a useful test bed to explore how different language models can collaborate with humans in a variety of social contexts.

Improv Theatre and AI

Improvised theatre explores how interesting narratives can emerge from establishing rules for simple social dynamics and rhetorical conventions. In contrast to scripted theatre, improv is built from spontaneity (Spolin and Sills 1963). Improvisers are trained to disengage executive cognition in order to allow their automatic responses to guide and justify a given and emerging social context (Johnstone 1979). Narratives emerge by assuming the presence of meaning. The performer only needs to accept offers: what is said and done on stage. There are no ‘wrong’ things a performer can say to invalidate the emerging narrative. For meaning to emerge for an audience however, each novel narrative statement must be followed up with some degree of agreement and justification. An improvisational scene is ultimately judged on the degree to which novel statements can be integrated back into the previous given circumstances. That process is called justification and is synonymous with an ongoing adaptation, by the actors—thrown out of their comfort zone—to the changing dynamics of an improvised narration. This practice is what makes improv theatre such a useful platform to explore the creative capacity of artificial intelligence. For the AI to perform ‘well’ it cannot simply introduce novel narrative subjects, but must also be able to adapt to the emerging given circumstances, akin to the desiderata of AI systems capable of generalising to unseen data.

Previously, (Mathewson and Mirowski 2017; 2018; Cho and May 2020) explored how an AI trained on movie or improv dialogue could generate interesting narratives as a per-
former within an improvised scene, and demonstrated that conversational agents built using recurrent neural networks or transformers, e.g., GPT-2 (Radford and others 2019), could indeed move a given narrative forward when human agents were operating to accept and justify the statements. In their setup, the AI only functioned as a character in a given scene. In our study, we examine how AI performs in the role of the narrator.

Methods

Datasets for AI improv

Before large language models, conversational agents were trained on datasets geared towards dialogue, like the Cornell Movie Dialogs Corpus (Danescu-Niculescu-Mizil and Lee 2011) and OpenSubtitles (Tiedemann 2009). The latter was used to develop conversational models such as (Vinyals and Le 2015) and the improv theatre-specific chatbot A.L.Ex (Mathewson and Mirowski 2017) from HumanMachine\(^2\). Later on, an improv-specific dataset of yes-and exchanges from improv podcasts was curated in (Cho and May 2020).

While employed in the context of improvised storytelling, our work departs from generating dialogue and focuses on storytelling from the perspective of a narrator. We hypothesized that the best datasets would come from general fiction novels as well as synopses and plot summaries. Coincidentally, Large Language Models (LLMs) are now pre-trained on comprehensive and diverse sets of corpora and are capable of memorising diverse linguistic patterns from books, novels, movie scripts, newspapers or blog posts (Radford and others 2019; Brown et al. 2020). From the perspective of thematic diversity and specificity, the need for collecting specific training data seems to have become less of an issue.

Live curation, mitigating of model bias

There are however trade-offs between the predictive power of large language models, and their embedded biases or their misalignment with desired societal values, which have been discussed in (Bender et al. 2021; Kenton et al. 2021).

Our approach to mitigating these biases and to the removal of offensive content relies on a combination of automated filters and human curation, performed in real time in the context of a live show. First, we remove sentences that contain known offensive words from a blocklist, and all generated sentences are validated using multiple filters for inflammatory, hateful or sexual content by the Perspective API\(^3\). Second, the human who operates the storyteller interface has agency in both how they formulate and type the context, and in what sentences produced by the AI they choose to read, with a possibility to omit or reword parts of those sentences.

Interactive live AI narration on stage

Our interface works in the following way: each time a sentence is typed by the operator, it is concatenated to the context of the scene. GPT-3 is then run 3 times on the whole context, thus generating three sets of sentences of total length up to 100 characters for each set. The operator has the choice of selecting none, one, or several of these sentences, in the order they choose\(^4\).

An important aspect of the human-machine interaction on stage is that the actors’ performance and the operation of the AI happen simultaneously, i.e. that the operator types context prompts and chooses AI-generated suggestions at the same time as scenes unfold. The human operator may then interrupt the scene, in a similar way to an improviser “editing” the scene. This delegates to the human cast and to the operator the artistic choices of timing—a crucial element of comedy—and maintains the liveness of the performance.

Story initiation

We initially experimented with a system for automated selection of initial writing prompts from the novel-first-lines-dataset (a crowdsourced dataset of first sentences of novels)\(^5\). The single-word audience suggestion would be matched with a fixed set of 11k sentences using sentence-level embeddings computed using the Universal Sentence Encoder (Cer et al. 2018) combined with approximate nearest neighbor search\(^6\). Early trials during improv rehearsals demonstrated that the first lines of novels were not informative enough for the actors performing much shorter scenes, and that the actors preferred to initiate the story themselves.

Avatar for the AI narrator

We designed a virtual avatar that personified the AI narrator. That avatar consisted of a 3D model of a robot, inspired by Aldebaran Robotics’ Nao\(^7\) and imported into Adobe Character Animator\(^8\) as a puppet controlled by facial expressions of the operator as they are reading the AI-generated lines. Instead of using computer-generated voice, we relied on human voice for expressive interpretation. The operator was

---

\(^1\)https://humanmachine.live
\(^2\)https://github.com/janelleshane/novel-first-lines-dataset
\(^3\)https://github.com/janelleshane/novel-first-lines-dataset
\(^4\)https://github.com/janelleshane/novel-first-lines-dataset
\(^5\)https://www.perspectiveapi.com/
\(^6\)https://www.perspectiveapi.com/
\(^7\)https://github.com/janelleshane/novel-first-lines-dataset
\(^8\)https://www.adobe.com/products/character-animator.html
standing behind the TV screen that was projecting the avatar on the stage.

Evaluating AI in performance
We worked with a small team of professional improvisers to build and rehearse an original 50 minute performance that included a series of short AI-assisted scenes followed by a 12-minute AI narrated long-form improvisational scene. We then presented 2 performances for public audiences. We present a partial transcript from one of the AI narrated performances and discuss how well the AI was able to offer contextually relevant suggestions that advanced the plot. We also administered anonymous surveys to the performers (p1-p5) as well as 9 audience members after each performance (a1-a9). The surveys consisted of a series of open ended questions regarding how they experienced the AI on stage. Our surveys were conducted in accordance with the approved ethical standards of our public research institution.

Results
We relate a 12-minute long-form improvisation between 6 actors. GPT-3 generated altogether 455 sentences of suggestions, but only a subset was selected by the operator. The following extract is an exact transcript of only the inputs or context given to the GPT-3 language model. **Sentences in bold correspond to GPT-3 outputs, which are fed back again as context.** (Notes in italic and between parentheses are scene descriptions that were not input to GPT-3). The transcript of performer lines and interactions is not included because of the difficulty of transcribing an improvised performance where many actors may speak on top of one another. The audience gave the suggestion: “Pizza Hut”.

At the Pizza Hut. Brian and his date lost patience. *(The operator misunderstood the relationship between the two protagonists.)*

There was always a reason for them to admire each other. Brian was an expert at making pizza. Sally found her vocation, making pizza like Brian. Brian started listing all the products... Baguettes, patisserie... Sally asked Brian for help. *(The operator made a confusion in the name, as it was Sandra, not Sally.)*

**The door opened and a burly man entered, followed by his wife.** *(A couple entered the pizzeria, the man spoke with a heavy voice.)*

The husband and the wife entered the pizzeria. They asked for suprèmes, with garlic bread. **Both women had crushes on Brian.** *(The unnamed wife briefly approached Brian.)*

Sally searched for pastries. *(The husband and the wife asked for vodka. (Unused suggestion.)*

They got creme patissiere... **Brian apologized.** *(Sally/Sandra was rolling pizza on the floor.)*

Sally was dreaming about becoming a master patissier. **She continued to look for pastries.** *(Sally/Sandra said she was done working at Pizza Hut and wanted to resign. Scene transition, with an angry boss entering the stage.)*

Brian’s boss told him he would let her go. Sally gave her notice. The boss refused. The boss was cruel. **Brian asked the boss for her resignation.** *(The boss made a mistake. (A confrontation took place between Brian and the boss, the boss later started behaving apologetically.)*

**Brian and Sally left the pizzeria.** *(A male actor stepped in to play the newly introduced Sally.)* *(Scene transition to Sandra at a restaurant owned by the burly man and his wife.)*

Sandra pursued her dream of being a pastry chef. Sandra was serving the old burly couple. **The burly man was impressed.** *(The burly man and his wife complimented Sandra.)*

Even though Sandra was violating safety regulations. **Sandra was getting tired.** *(Scene transition to the boss joining the group.)*

The boss came to apologise to Sandra. Sandra said that she remembered him. He was diminished. He was wondering if it was safe to do it on the floor... She heard about Brian. Can you come back, he asked. The boss was apologetic. **Sandra thanked the boss, who helped her. Brian and Sandra were both happy.**

Sandra was proud. The boss was really clear. **The boss was jealous.** *(Scene transition to the boss joining the group.)*

As the 12-minute scene unfolded, the operator was typing a summary of it as inputs to an interface to GPT-3. For each line of context that was input by the operator, there were many alternative suggestions that could have been selected, and this transcript shows only the ones that were actually chosen and presented to the cast and to the audience. The decision to intervene in the narration and the timing and delivery of each intervention were choices made by the operator, who was simultaneously voicing and animating the virtual avatar, as well as observing the live improvised scene.

Just like in the first show (for which we do not report the transcript in this paper), the AI-assisted narrator’s interactions became more frequent as the scene was unfolding and the characters established. The motivation for this was to let the actors establish the characters and their relationships first, and to start intervening only once the cast had an initial guess of the narrative arc of the story.

Audience and performer response
We provide the following small sample of 9 audience responses as useful observations to guide discussion rather than evidence of findings that can be generalised. 7 of the respondents indicated the presence of AI itself as the most significant motivational factor in attending the events. 6 reported overall satisfaction with AI narration, 1 reported neutral satisfaction, and two reported dissatisfaction. The AI narrated scene was the most frequently cited (5/9) response to the question ‘What did you enjoy most about the show.’

All 5 performers reported satisfaction with the ability of the AI to move the story forward. 3 however also ‘slightly
agreed’ that the AI ‘mainly introduced absurd or random information’ into the scenes. We present the following quotes from performers about their experience to advance discussion about the relationship between the insertion of surprising plot points experienced as both ‘random’ and useful in advancing the story arc.

- As a performer I had to physically become the character (the AI) described in the narration. This pushed me to a certain pov / voice/ physicality which I probably wouldn’t have chosen i.e. a gruff, muscly patisserie store owner. (p1)
- (The AI) added a level of randomness and craziness different from a human brain. (p2)
- (The AI) really helped the plot move forward, but without being too prescriptive, and enabled me to focus on character development, relationships, emotions and object work. (p3)
- I did a few scenes as the protagonist where I was sad, and then (the AI) would say ‘she was happy’ or similar, but I loved that as I have to justify it and it was funny! (p4)
- Generally the narrative direction (of the AI) helped the show move forward in a good direction (p5)

Discussion

In the above exchange between the operator’s inputs and the AI suggestions, one can notice that the AI introduced two key characters (the burly man and his wife) who played the role of mentors for the main protagonist, Sandra, and enabled the resolution of the story by complimenting Sandra’s work. The AI’s suggestions also satisfied a classical narrative arc by allowing her “dream to come true” and achieving her transformation into a “great pastry chef”. This illustrates the capacity for an AI-based narrator–operating in tandem with a human curator who makes timing decisions–to generate novel and meaningful plot points.

Interestingly, as (p4) noted, the AI-provided suggestions did not consistently keep the affect or motivational stance of some characters (e.g., the boss was first cruel, then apologetic and even helping Sandra). Where this inconsistency might invalidate a progressing story when uttered from a character (and subsequently fail a Turing-test), in the mouth of a narrator it can encourage performers to maintain classical story arcs that require characters to change and adapt over time (Aristotle 350 BC). The inconsistencies of the AI-generated text were interpreted by the cast as narrated reversals of feelings, and challenged the performers (as p1 suggests) to allow themselves to change and be affected by each other in surprising but meaningful ways. In improv theatre this is described as a ‘status’ reversal where the ‘low’ status of a character at the beginning of a scene becomes ‘high’ status by the end (Giebel 2019). Such reversals are in practice often difficult for human improvisers to execute, as one instinctively attempts to maintain their given status or fight to maintain ‘high’ status. In this instance, the AI drove the plot more aggressively forward and motivated the performers to shift and adapt status to the evolving circumstance that in effect provided a more clear beginning, middle, and end to the story.

As a creative partner, rather than simply providing strange or absurd plot points to challenge the human performers to make sense out of, the AI seems to have removed some of the cognitive load for improvisers (as with p3) allowing them to concentrate on relationships. Without a narrator, improvisers must both react spontaneously in the moment, and remember to engage narrative techniques such as status changes to move the story forward. This important practice of narrative making can be understood as “a carefully argued process of removing and adding participants” (Kumar et al. 2008). The practices of theatrical improvisation and acting techniques such as Meisner actor training (Moseley 2012) explicitly ask performers “not to be in their heads”, meaning not to withdraw from the live performance in order to plot or to reflect and comment on the scene, but rather to dedicate their entire attention to what is happening in front of them on the stage. We believe that one of the potential applications of computational creative systems could be to alleviate the cognitive load of performers to shift their focus from plotting to reacting.

Strikingly, the seemingly random characters introduced by the AI were often the result of human error. But even when such errors were introduced the performers playfully accepted the offers that resulted in comic relief, skilfully transforming an error into a serendipitous opportunity to ‘break the fourth wall’ and to connect with the audience. For instance the wrong naming of the main protagonist (as the operator mistakenly and repeatedly typed “Sally” instead of “Sandra”) led the improvises to quickly introduce, then shelve, a temporary character. This tight collaboration between improvisers and AI prevented the introduction of a new character that may have otherwise been considered a ‘less carefully argued’ addition to the narrative, to still perform a useful function (comic relief) without disrupting the evolving story.

Conclusions

Narrative theatrical performances encapsulate human culture, social interaction, physical expression and natural human emotion. Improv is an ideal test-bed to explore questions about the human-AI collaborative creative capacity. It has been proposed as a grand challenge for artificial intelligence (Martin, Harrison, and Riedl 2016). We believe that AI-as-collaborator, as in this current study, uplifts artists, as opposed to challenging them.

Language models capture statistics of written corpora of human culture, and thus provide human audiences with a mirror of typical narrative tropes and biases. Thus, they highlight the need for human interpretation and curation of AI-generated content. Our two-pronged approach of automated filters followed by human operator selection of sentences, illustrates a transfer of responsibility from the language model to the (human) narrator—not unlike a typical improv show, where the human cast are responsible for the story they tell (e.g., “punching up, not down”) and adapt to their audiences (e.g., family-friendly vs. late-night shows).
This work is, to the best of our knowledge, the first staging of an AI narrator co-creating improvised theatre alongside humans for a live audience. Timing and aesthetics are significant factors for the human experience of AI by audiences and cast members. The ease of use of the narrative interface for the human operator impacts how quickly they can add to the language model context or choose from its outputs. Finally, the imagined ‘personality’ of the AI narrator play a role in co-creation. These are important avenues for future research on human-AI co-creation.

Acknowledgments
The authors wish to acknowledge the anonymous audience participants who filled the questionnaires, and above all, to thank the actors from improv theatre troupe Improbotics who collectively created the artwork that enabled this specific study, namely Harry Turnbull, Julie Flower, Marouen Mraihi, Paul Little and Sarah Davies.

References
Aristotle. 350 B.C. Poetics. Translated by S. H. Butcher.
Giebel, J. D. 2019. Improvising tactical choices based on status or “who’s driving the dramatic action bus?”. In Objectives, Obstacles, and Tactics in Practice. Routledge. 141–150.
3. Musical Creativity
MuSyFI - Music Synthesis From Images

André C. Santos¹, H. Sofia Pinto¹, Rui P. Jorge², Nuno Correia³

¹INESC-ID, Instituto Superior Técnico
²CESEM, FCSH, Nova University of Lisbon
³NOVA LINCS, FCT, Nova University of Lisbon

andre.carvalho.dos.santos@tecnico.ulisboa.pt, sofia@inesc-id.pt
ruijorge@fcsh.unl.pt, nmc@fct.unl.pt

Abstract
MuSyFI is a system that tries to model an inspirational computational creative process. It uses images as source of inspiration and begins by implementing a possible translation between visual and musical features. Results of this mapping are fed to a Genetic Algorithm (GA) to try to better model the creative process and produce more interesting results. Three different musical artifacts are generated: an automatic version, a co-created version, and a genetic version. The automatic version maps features from the image into musical features non-deterministically; the co-created version adds harmony lines manually composed by us to the automatic version; finally, the genetic version applies a genetic algorithm to a mixed population of automatic and co-created artifacts.

The three versions were evaluated for six different images by conducting surveys. They evaluated whether people considered our musical artifacts music, if they thought the artifacts had quality, if they considered the artifacts `novel`, if they liked the artifacts, and lastly if they were able to relate the artifacts with the image in which they were inspired. We gathered a total of 300 answers and overall people answered positively to all questions, which confirms our approach was successful and worth further exploring.

Keywords: Computational Creativity, Inspiration, Feature Translation, Genetic Algorithm, Music Generation

Introduction and Motivation
MuSyFI tries to model an inspirational creative process by automatically and semi-automatically generating music from images. Having chosen images as source of inspiration, any image, it generates music that can be perceived as being related to it. This relationship is subjective, since there is virtually an infinite number of musical artifacts that can be generated from an image. We do not target a sonification endeavour as we do not merely translate visual features into musical features to produce sound, but rather attempt to model a possible creative inspirational process whose starting point are images.

We aimed that our musical artifacts could be considered creative, aesthetically pleasing, and that the music could be relatable to the images that inspired them. Each one of these goals is subjective, which makes evaluation harder. This was positively evaluated through questionnaires answered from 300 respondents.

The rest of the paper is organized as follows: We start by reviewing related work. Next, we discuss feature extraction from images and describe the main features extracted. We then explain in detail the pursued visual to music mapping. Afterwards, we describe the genetic algorithm developed in this work, along with all its processes and the tests we did to validate our parameter choices. We present and discuss our results, and conclude the paper with both a critical summary as well as some indications for future work. Our musical artifacts and respective images can be seen and heard on our website¹.

Related Work
There are several systems that generate music computation-ally - EMI (Cope 1989), GenJam (Biles 1994), MuseNet (Payne 2019), to name a few - using different approaches - from Knowledge-Based Systems (KBSs) to Artificial Neural Networks (ANNs). However, few try to model inspiration. In one such work, Horn et al. (2015) extract the dominant colours from an image and shape a 3D vase according to those colours, effectively implementing an inter-domain mapping of features.

Later, Teixeira and Pinto (2017) generated music inspired in images, outputting three versions: raw, harmonized and genetic. For the raw version, the image is divided into quadrants which are then mapped to measures. The colours in each quadrant determined the notes and the chords played in each measure. The notes were chosen from a diatonic scale, and chords were major or minor, depending if the colour was warm or cold, respectively. The rhythm was picked from 44 drum patterns from a database, based on the emotion state of the quadrant given by its visual characteristics. The harmonized version adds a bass line to the raw version, and limits chords to three or four for the whole artifact, obtained from the whole image. Likewise, only two drum patterns are used in this version. Finally, the genetic version applies a GA to a population of 24 individuals (the raw version, the harmonized version, and 22 other variations thereof). The GA’s fitness function is a combination of seven different criteria.

¹http://web.tecnico.ulisboa.pt/ist178488/
Our work addresses the same problem as Teixeira and Pinto but takes a very different approach. For example, the authors divided the image into a grid of quadrants and analysed the image quadrant by quadrant, whereas we divide the image into its saliencies and non-salient background. Additionally, Teixeira and Pinto use pixel based features, whereas we also added shape and position related features.

Both approaches produced interesting and promising results. The difference lies in the path chosen to arrive at the same goal. Both paths are valid and both paths lead us through interesting landscapes, which, by its very nature, attest to the subjectivity involved in this approach and the weight personal aesthetics has in the final result.

**Image Feature Extraction**

To use images as an inspiration source, we first need to extract features from them to map these into musical features. In other words, we need to process the image.

**Saliencies**

Saliencies are features that draw attention to us when looking at an image or a series of images and saliency detection is an active research subfield of computer vision. OpenCV (Bradski 2000) has a saliency detection module with two static saliency detection algorithms. One of those algorithms is the StaticSaliencyFineGrained (Montabone and Soto 2010). The authors based themselves on center-surround differences our eyes use to identify saliencies in images. An example of the saliency map obtained from this algorithm is shown in the second image of Figure 1, next to the original image. As can be seen, either the dog or parts of it are identified as being salient, as well as some parts of the grass. However, the dog is not identified perfectly as a whole.

To improve upon this result, we used another image processing algorithm, GrabCut (Rother, Kolmogorov, and Blake 2004), also implemented in the OpenCV library. It segments the image into foreground and background homogeneous regions. The GrabCut algorithm usually needs a human to indicate where the background and foreground are. However, by using the saliency maps, we can bypass the human input and still obtain accurate and autonomous results.

If we classify each pixel in the saliency map as one of the algorithm’s four possible values, we can then feed the image to the algorithm and use its output to obtain the correct saliencies of the image. We can observe this process in Figure 1.

**Contours**

A contour is a curve that joins all the continuous points along a boundary which encircles a region of pixels that have the same colour or intensity. Contours were used to study the shape of the saliencies we extracted. We used the `findContours` function from OpenCV which receives a binary image as input and finds its contours. It is based on

\[
\text{findContours} \text{ function from OpenCV which receives a binary image as input and finds its contours. It is based on}
\]

\[
\text{Suzuki 1985) and it works by applying border following to the binary image, labeling the borders it finds. Here, border and contour are used interchangeably.}
\]

We then find the contour’s centroid and plot the distance to the centroid along the border, starting from the minimum distance point and continuing along the border, counterclockwise. This contour distance plot can be seen in Figure 2 with respect to the dog image of Figure 1. The y-axis represents the distance to the centroid relative to its maximum value, and the x-axis represents the number of pixels along the contour. This means each plot always has a peak of value 1 and that, since bigger saliencies have bigger contours, bigger saliencies generate larger contour plots as well. The triangles in the plot are explained in subsection Melody.

**Colour**

We use both Hue, Saturation, Value (HSV) and Hue, Saturation, Lightness (HSL) colour models to extract what we call the dominant colours of an image. We divide the 360 hues into 12 main hue bins. If a hue bin has at least 10% of all the image’s pixels, then it is considered a dominant hue tone. The 10% value was obtained empirically, and while it might seem low, it allows us to retain important colour information about the image that would otherwise be lost. We can then plot the dominant colours histogram. In Figure 3 we show the histogram for the original dog image. The y-axis represents the number of pixels and the x-axis represents the respective hue bin. All hues whose bars are higher than the black horizontal line (representing the 10% threshold) are dominant colours. We should note that, for each bin, we have two overlapping bars: the first one represents the respective hue with max saturation and max lightness, and the second one represents the same hue but with the average saturation and average lightness extracted. We do this to have an idea of the pure hue and an approximation of the actual hue that is present in the image. In Figure 3, an example of this are the two bars representing the main hue marked
Feature Mapping

Having presented the visual features and how we extracted them, we now explain the visual to musical feature mapping we conceived by explaining how general features of our artifacts were defined, as well as how the melody and harmony for our musical artifacts were pieced together.

General Composition Features

Before defining the melody and harmony parts of our musical artifacts, we first define their time signature, tempo, and key/scale.

The most common time signature in music today is $\frac{4}{4}$, while other time signatures (like $\frac{6}{8}$ for example) are usually used to compose more complex musical pieces, so we decided to define the time signature for all our musical artifacts as being $\frac{4}{4}$ as well.

Regarding the tempo of our musical artifacts, we chose to associate it with the number of edges in an image. Images with more edges seem, in our opinion, more frenetic and having a faster pace than an image that does not have as many edges. We defined a minimum tempo of 60bpm and a maximum of 150bpm since they are relatively slow and fast tempos, respectively. We apply the Canny edge detector algorithm, count the number of non-zero pixels (edge pixels), and divide them by the total number of pixels in the image, obtaining what we call the edge ratio. This ratio is then divided by 0.3 to normalize it, since that was the maximum observed edge ratio in our dataset with our parameter choice. We use this new ratio to define the tempo of the song, being that 0 corresponds to 60bpm, 1 corresponds to 150bpm, and other ratios are distributed linearly according to Equation (1).

$$\text{tempo} = \lfloor \frac{\text{edge ratio}}{0.3} \times (150 - 60) + 60 \rfloor$$ (1)

Finally, regarding the key, we decided to generate tonal musical artifacts so our pieces have a tonal center with which a diatonic scale is associated with. To choose the tonal center of our scale, we used color. We extract the most dom-

Edges

Edge detection is a classical image processing problem that tries to identify points in an image where brightness changes abruptly. The set of these points forms a set of curved lines called edges. John F. Canny developed a staple algorithm for edge detection that was eventually named after him, the Canny edge detector (1986) which is implemented in OpenCV’s Canny function. The high and low thresholds were defined empirically and subjectively as 30 and 200, respectively, and we used the same thresholds for every image in our dataset.
inant hue tone of the image, and use the association of Figure 4, where we overlap a 12 hue tone circle with the Circle of Fifths\textsuperscript{4} since similar hue tones harmonize well with each other (ex.: red and orange) as well as adjacent notes in the Circle of Fifths (ex.: C and G). We use 12 hue bins so as to make a direct association between colours and semi-tones. We should note that the first association made was that red be associated with A, since 440 Hz is the standard tuning pitch and corresponds to the A tone, and in the visual spectrum, 440 Hz corresponds to the colour red, which is its first visible colour.

![Circle of Fifths](image)

**Figure 4:** Colour and tone association according to the Circle of Fifths.

To define whether the scale chosen is major or its relative minor, we turned to the average Value (from the HSV colour model) that dominant colour has. If it is lower than or equal to 0.5, the scale chosen is the minor one. If it is higher than 0.5, the scale chosen is the major one. This was done because major scales sound "brighter", while minor ones sound "darker".

### Melody

Melody is usually what stands out in a song. With this in mind, we decided to associate the melody part of our musical artifacts to the saliencies we extracted.

We wanted to use the shape of our saliencies and map it in some way into the melody of our musical artifacts. We associated more angular shapes to higher-pitched sounds, and flatter shapes to lower-pitched sounds. A similar association was studied by Ramachandran and Hubbard (2001). Furthermore, sharper shapes give, in our opinion, a bigger sense of urgency and speed when compared to rounder shapes. So we associated the first type of shapes with quicker notes strung together, and the second with slower, longer notes. We can see different shape examples in Figure 5.

To measure the different types of shapes, we find the saliencies’ contour plot peaks, fit triangles onto them, and then measure shape properties through the triangles. To be able to draw triangles onto the peaks, we need three points for each one. The first point is the peak point itself. The other two are the baseline points, for which we first need to define the baseline value of the triangle. This is done by finding the halfway points between the peak and the previous/next peaks in the contour. Then, we define the baseline value as the median value between these two halfway points. Having the baseline value (fixed $y$), we just need to find the $x$ coordinates for the triangle’s baseline points. We do this by intersecting the baseline and the contour plot and finding, to the left and to the right of the peak, the points whose values are closest to the baseline to form the triangle\textsuperscript{5}. We define the neighbourhood where we search for these points as being $2 \times (peak.value - baseline) \times contour.size$ to both sides of the peak, where $peak.value$, $baseline$, and $contour.size$ represent the ordinate of the peak, the ordinate of the baseline, and the number of pixels of the contour, respectively. Then we minimize the error for each of the neighborhood’s points $(x, y)$, given by Equation (2):

$$error = \left| baseline - y \right| \times 1000 + \left| peak.x - x \right| \times 0.06 \quad (2)$$

$peak.x$ is the abscissa of the peak. The weights, 1000 and 0.06, were attributed empirically. After drawing the triangles onto the contour distance plot, we obtain a plot as in Figure 2.

Each triangle maps to a single note and we add notes onto the musical sheet sequentially from left to right. The first note corresponds to the minimum contour distance point.

To turn the triangles into notes, we need to define the notes’ pitch and duration. For the pitch, we calculated the peak’s angle using trigonometry. We fit a whole scale between the minimum and maximum angles of the contour plot, and every note in between is uniformly distributed, with more acute angles representing higher notes and vice versa. Also, our angle to pitch distribution is not deterministic. To generate more diversity between notes, but still choosing a similar note to the one picked, we fit a gaussian around the chosen note with a standard deviation $\sigma = 3 \times \frac{\text{repeated note}}{2}$, where $\text{repeated note}$ is the number of times that note is chosen consecutively.

\textsuperscript{4}The Circle of Fifths is a musical tool that depicts the relationship between the 12 different tones on the chromatic scale.

\textsuperscript{5}The contour plot is not continuous, so we need to find the closest point of the intersection and not the exact points.
For the duration, we calculate the triangle area to contour distance’s integral ratio over the triangle’s baseline points. We rounded the ratios to the decimal point and, since higher ratios means a good fit between the triangle and the peak, i.e., a sharp peak, and a lower ratio a rounder peak, we defined that whole notes correspond to ratios rounded to 0.0, half notes correspond to ratios between 0.1 to 0.4, quarter notes correspond to ratios between 0.5 to 0.7, eighth notes correspond to ratios rounded to 0.8, and sixteenth notes correspond to ratios rounded to 0.9 and 1.

Rests were also defined from contour distance plots. We measure the relative distance between peaks - that is, we count the number of pixels between peaks, and divide them by the total number of pixels of the contour - and then round it to the decimal point. These relative distances are usually very small, so we associated them with rest durations as follows: if the relative distance is rounded to 0.0, no rest is added between notes; if it is rounded to 0.1, an eighth-note rest is added; if it is rounded to 0.2, a quarter note rest is added; every value higher than that corresponds to a half note rest being added between notes.

To define the octave of our melody tracks, we use the saliency’s most dominant colour average Lightness (from the HSL colour model): the higher the lightness, the higher the octave and vice versa. We defined the range of possible octaves from C0 to C6. Then, we divided the range of possible Lightness values into seven different bins, classified the average Lightness into one of these bins linearly, and directly associated lightness bins with octaves.

Regarding the timbre of the saliencies’ melody lines, we decided to use the most dominant colour of each saliency - since timbre is also known as tone colour - and we mapped different hue tones to different families or groups of instruments. This association is completely subjective and could have been done in many different ways, but we tried to arrange families so that neighbouring families were associated with similar colours. Since Lightness determines the octave of our tracks and for the same Hue we can have very different Lightness values as the two colour channels are independent, Hue only determines the instrument family and not the instrument itself. Accounting for Lightness, our association is presented in Table 1. Minimum and maximum Lightness values, which roughly correspond to black and white, were associated to Timpani and a Piccolo Flute since they are low and high register instruments, respectively.

Finally, each saliency inspires a melody line, so when an image has more than one saliency, multiple melody lines are generated. They are played radially, that is, they start sooner if their respective saliency is closer to the center of the image and vice versa.

Harmony

We associated harmony to the non-salient background, i.e., the image that remains when we remove the saliencies. We analyse the non-salient background as a whole, defining one harmony track per image. First we extract the non-salient image’s dominant colours, and we use the same association of Figure 4 to define the tonality of the harmony track’s chords. However, we define for each tonality five different types of chords: major and minor chords, augmented and diminished chords, and power chords.

If the chord’s dominant colour has a Lightness of 0.9 or higher, the chord is associated with an augmented chord; if it is lower than 0.1, it is associated with a diminished chord. If its Saturation is lower than 0.25 (with its Lightness between 0.1 and 0.9), the chord becomes a power chord. If none of the cases above happen, the colour is associated with a major or minor chord. In that case, the type of chord is defined as follows: if the Value of the dominant colour is 0.5 or lower, the colour is associated with a minor chord; if it is higher than 0.5, a major chord is used.

We assigned one chord to each of the musical artifact’s measures. The chord for each measure is chosen according to the dominance of its dominant colour in the image: more dominant colours are more likely of being selected and vice versa. The range of possible octaves for the harmony track is between C2 and C5, inclusively. To assign it, we calculate the median octave between the melody tracks of the artifact and subtract one to this value.

For the timbre, we tried to measure if an image used colour tones close to each other, or colour tones that contrasted each other. We calculate the relative distance between each dominant colour and the most dominant colour of the image around the colour circle, and calculate the average colour distance for the whole image. Then, we assign the harmony track’s instrument by picking the longest track’s dominant hue as the hue center, and then traversing that distance in the colour circle in a clockwise or counterclockwise fashion (randomly picked between the two) to decide the harmony track’s hue, and, consequently, the harmony track’s instrument, using our hue to instrument association from Table 1.

If an image has only one melody track, its harmony track is comprised of a chord line, that is, at each measure all notes from its previously defined chord are played. If the image has several melody tracks, the harmony line consists only of a bass line, so only the tonic of the chord is played at each measure.

Two versions are presented at this stage: an automatic version and a co-created version. The automatic version is obtained using the mapping we explained in this section, with harmony lines solely comprised of whole notes. The co-created version stems from the automatic version, but its harmony track is selected from a set of manually composed harmony tracks. In total, we composed 24 harmony lines: one bass line and one chord line for each family of instruments from Table 1. The chord or bass line is chosen according to the instrument family.

Genetic Algorithm

Up until this point, our work can be interpreted as feature translation between domains and, although we believe our feature mapping is novel and rich, feature mapping alone does not model any creative process. While we could argue it possibly models the underlying inspirational process of
a creative process, we need something more to model the exploratory component of creativity. With this in mind, and also to try to improve upon our results, we use a genetic algorithm.

**Structure**

We first generate the initial population by using our feature mapping $n$ times to produce $n$ musical artifacts, where $n = 100$ denotes our population size. We use a mixed initial population of 50% automatic musical artifacts and 50% co-created artifacts. Our Selection step is standard, with an elitism factor of 25%. Then, each pair of individuals selected has a 90% chance of being crossed over and each one has an 80% chance of being mutated. We continue selecting individuals for 300 iterations after which we output the fittest individual, which represents our genetic version.

**Validation**

A study was conducted where we observed how the fitness function evolved to select the values for population size, number of iterations, and percentages of crossover and mutation. We first tested different population sizes - 25, 50 and 100 - and we observed that the fitness of individuals increased the more we increased this number, but we settled for 100 since a larger number made the program slower. Next, we tested the number of iterations, starting with 50, then 100 and finally 300. We observed significant improvements over the fitness values of individuals, but a stagnation at around 300 iterations, so we fixed that parameter at that value. Finally, we initially set our crossover and mutation percentages as 90% and 80% respectively since both mechanisms are extremely important in the evolution of our musical artifacts. We then dropped each one individually to 10% and observed that the fitness values obtained were worse, so we kept the initial parameters of 90% for crossover and 80% for mutation.

**Crossover**

Crossover happens between a pair of individuals and it always involves half of each musical artifact’s measures, although not necessarily the same half each time.

There are three different types of crossover that can happen: melody track crossover, harmony track crossover, and mixed crossover. The first type happens between the melody tracks of the two selected musical artifacts. Harmony track crossover is identical, but between the artifacts harmony tracks’ measures. Finally, mixed crossover combines both previous types of crossover, switching both the chosen melody track’s measures, and the same harmony track’s measures across two musical artifacts.

The crossover step is crucial since it is what allows for a mixed harmony line in our genetic version. We use the fitness function’s last criterion to try to group the genetic version’s harmony line into uniform groups of the other versions’ harmony lines. Hence, the genetic version’s harmony line tends to switch between the two periodically.

**Mutation**

Mutation can happen to any selected individual. In a mutation, one feature of the selected individual is changed. In our GA, we defined six different types of mutations: note duration, note pitch, note switch, chord type, chord pitch, and melody track instrument.

The note duration mutation changes the duration of a randomly selected note, pitch mutation affects its pitch, and note switch mutation simply switches two contiguous notes. Chord type mutation changes the type of a randomly selected chord to another type (M, m, Aug, Dim, or PC), and chord pitch mutation changes its tonic to another pitch from the pitches associated with the dominant colours of the non-salient image. Finally, melody track instrument mutation simply mutates a randomly selected melody track’s instrument to one of its neighbouring colour’s instruments according to our association. The only type of mutation with a different probability of occurring is the melody instrument mutation with only a 1% chance of happening. Otherwise, the different mutations are distributed uniformly, each having a $99%/5 = 19.8\%$ chance of happening.

**Fitness Function**

The fitness function evaluates how fit individuals are. In other words, it defines how “good” or “bad” individuals are, according to some criteria established *a priori*. It is usually defined as a set of criteria that optimize a function, but since there are no optimal musical compositions, the criteria we present here are subjective, even if based in music theory.
concepts. With $f$ being the final fitness value, we defined our criteria as follows:

- If the musical artifact starts or ends with the tonic chord: $f ← f + 100$.
- If the musical artifact starts or ends with a note that belongs to the chord of that measure: $f ← f + 100$.
- Every time the underlying chord appears in three consecutive measures: $f ← f − 60$. If the harmony line is played by a bass line instead of a chord line, we check the tonic itself.
- For each melody track and for each of its measures, when there is a note on the strong beat: $f ← f + 10 \times \text{note.duration}$, where note.duration denotes the duration of the note.
- For each melody track and for each of its measures, when its strong note belongs to the measure’s chord: $f ← f + 40 \times \text{note.duration}$.
- For every melody track note that does not belong to either its respective chord or its respective scale: $f ← f − 60$.
- For each melody track, if each of its measures has its respective chord notes: $f ← f + 100$ per chord note.
- If a melody track’s measure has no notes that belong to its respective chord: $f ← f − 200$.
- If an interval between melody track notes is bigger than 12 semi-tones: $f ← f − 2 \times \text{pitch.difference}$, where pitch.difference is the difference between the notes’ pitches.
- If there are multiple melody tracks that are played by the same instrument: $f ← f − 50$ per repeated instrument.
- Finally, $f ← f − 200 \times$ the standard deviation from the groups of co-created vs automatic harmony lines. At the start of the GA, since all musical artifacts only have groups of one type: $f ← f − 200 \times \text{number.measures}$, where number.measures denotes the number of measures of a musical artifact.

The overall fitness value of an individual is the linear combination of the different criteria values. Some of these criteria were given stronger predominance and hence higher values. These values were defined subjectively and empirically, and are relative.

**Evaluation**

The hypotheses we wanted to verify were if people thought our musical artifacts had quality, if they thought they were novel, if they enjoyed listening to them, and if they could relate them to their respective images. Since all of our goals are subjective, to evaluate our results we decided to survey people. The hypotheses questions were evaluated with a Likert scale from 0-5.

**Evaluation methodology**

We evaluated our system for six different images. The images were selected in order to have a balance between more abstract and more concrete images, images with different colour spectra and edge predominance, as well as images that had only one predominant saliency, images with several saliences, and images with no apparent salience at all. The images selected can be seen at http://web.tecnico.ulisboa.pt/ist178488/ and they include an Elf image, a Dog image, Rothko’s Green and Maroon, Pollock’s Blue (Moby Dick), Picasso’s Girl Before a Mirror, and Mondrian’s Composition No.10.

For each image, we generated three different musical artifacts, or versions - an automatic one, a co-created one, and the genetic one. That means that in total we evaluated 18 different musical artifacts. We split the six different images between four different surveys, two images per survey, one of the surveys repeating two of them, but paired differently. The surveys are identical in terms of the questions asked and their structure, the only differences were the images and the musical artifacts presented in each one.

Regarding the structure of each survey, we first profile the respondents by age, gender, degree of musical knowledge, and music genre(s) they are most familiar with or prefer best. Both the degree of musical knowledge and music genre preference are asked to see if we could find any patterns among people of the same groups, regardless those categories, but only the former was relevant as discussed in the Result Analysis. The other demographics did not amount to any relevant findings either.

Then, we present people one of the three different versions for one image. We first ask people if they think the sound sample has quality, if they think it is novel - with a description the term -, and if they enjoyed listening to the sound sample. Only after having answered these questions we present the respondent with the image from which the musical artifact was generated. We then ask if they relate the sound sample to the image shown. These questions are evaluated using a Likert scale of 0-5. We do not tell the respondent the sound sample was inspired by the image. Then we repeat this for each of that image’s other versions. The order in which we present the different versions is different for each image. Since we have three different versions, the possible ordered arrangements we can make with them are exactly six (matching the number of different images we chose to evaluate our system with), so we choose a different permutation for each image. This was done to avoid order bias, i.e., preventing people from getting too familiar and not answer each musical artifact’s questions independently from the other ones.

We also decided not to limit our respondents to people who did not know a priori about our project to try to obtain a larger number of answers and to be able to study for any bias regarding their previous knowledge of our musical artifacts being made by a computer. We asked people this at the end of each survey. The surveys were mainly distributed via social networks.
Result Analysis

Analysing the answers as a whole, we obtained exactly 300 answers in total, of which 87% of the respondents said they considered our musical artifacts as music, with only 13% saying they did not. Regarding the four main questions asked (Figure 6), generally speaking results are fairly positive across all four questions, particularly regarding quality where the most answered value is 4.

![Graphs showing assessment results](image)

Figure 6: Assessment of our main hypotheses.

Next, we separate people who knew about the project (106 answers) from people who did not (194 answers). In general, people who knew about our project tend to lower their expectations and evaluate our artifacts as being better, and the opposite can be said about people who did not know anything about it. A note should be made about musicians or people with a degree in music however: they are the complete opposite, i.e., those who know about our project give our artifacts the worst scores, but those who do not evaluate them extremely positively. However, we should also point out that in total we had 12 answers for this category, 6 of people who knew and 6 of people who did not, so, while it is an interesting hint worth further exploring, we acknowledge that personal bias might be a factor here with such a small sample size. We also divided the overall results by version (Figure 7) and we can see the genetic versions were generally preferred.

![Graphs showing assessment for different versions](image)

Figure 7: Assessment of our main hypotheses for the three different versions of our system.

There is only one version whose average quality values are below 2.5 (the neutral value, considering the Likert scale used), all the other averages - including all novelty ones - are above this value. Taking these results into account, as well as Boden’s creativity theory (2009) stating that creativity is some sort of combination between value and novelty, we can affirm that all our artifacts can be considered creative from a qualitative point of view. We should point out that a person who does not like a particular genre (subjective point of view) might still acknowledge that a particular song has quality, from an objective point of view. In that sense, we chose to ask if the artifacts had quality instead of what value they had.

Discussion

Recalling our goals once again, we aimed at generating musical artifacts that could be considered music and creative, aesthetically pleasing, and that could be related to the images in which they were inspired. After analysing results, we can safely say that our goals were met: the surveyed people generally considered our artifacts valuable and novel - and hence creative -, they generally enjoyed listening to them, and considered them to be music. Lastly, although results were more polarized, there were still more overall positive answers than negative ones regarding the relation between images and their respective musical artifacts. This is only natural since such relation is highly subjective. We can also conclude from the surveyed data, combining the answers for all the images, that our genetic version appears to be preferred over the other two versions, which also legitimizes its implementation.
Conclusions and Future Work

MuSyFI is a computer program that takes inspiration from images and is capable of generating three different versions of musical artifacts: an automatic version, a co-created version, and a genetic version. We wanted to model an inspirational exploratory creative process, therefore we used images as a source of inspiration and we implemented a genetic algorithm.

For the automatic version, a feature translation was established according to subjective and empirical criteria. We did not aim at making direct mappings or any sort of sonification, choosing instead indirect and innovative methods of feature translation across domains. We also added simple harmony lines composed by us to make the co-created versions, effectively making our program a co-creative endeavour as well.

Our GA combines both automatic and co-created versions into one genetic version. The musical artifacts generated are fairly interesting and we can notice the influence both have on the final genetic version. We should note that, while all genetic versions' artifacts follow a certain aesthetic guideline which is noticeable (defined by the GA’s fitness function), they still differ considerably amongst themselves inside that conceptual space. Furthermore, our genetic algorithm adds both cohesion and diversity to our musical artifacts by guiding them through the evolutionary process according to a fitness function, which in turn is based on music theory concepts.

All three versions were evaluated across six different images for a total of eighteen musical artifacts evaluated. Our goals were to be able to generate musical artifacts that could be considered creative, that could be considered music and aesthetically pleasing, and that could be related to their respective images. To evaluate our musical artifacts on such subjective goals, we surveyed people and asked their opinions. We gathered a total of 300 answers, which allowed a thorough evaluation of our musical artifacts. Results were fairly positive, with most people answering favourably to all the hypotheses we set out to validate, inviting further exploration of different methods of generating music from images.

That said, much work can still be done, both from an image processing perspective - trying to extract semantics from images or dealing with saliencies in another way for example - as from the music generation point of view - adding motifs and structure to the musical artifacts, and using other sounds than just MIDI sounds, to name a few.

Trying to combine machine learning techniques with a genetic algorithm could also be worth exploring further. Since machine learning techniques are usually good at finding patterns, we could try to find what “patterns” a song usually follows - what type of melodies, structure, chord progressions, rhythmic sections are used, etc. - and in that way try to measure its value, and then feed those outputs to a genetic algorithm with which we would try to add more diversity and novelty, exploring different possible outputs on top of those musical artifacts. There is also space for improvement regarding our GA, namely coping with a multiple objectives fitness function, or using a different evaluation framework like SPECS (Jordanous 2012), for example.

Acknowledgments

This work was supported by FCT project UIDB/50021/2020.

References


Supporting Computational Music Remiking with a Co-Creative Learning Companion

Erin J.K. Truesdell1, Jason Brent Smith2, Sarah Mathew1,
Gloria Ashiya Katuka3, Amanda Griffith4, Tom McKlin3,
Brian Magerko1, Jason Freeman2, Kristy Elizabeth Boyer4
1Expressive Machinery Lab, Georgia Institute of Technology, Atlanta, GA 30308, USA
2Center For Music Technology, Georgia Institute of Technology, Atlanta, GA 30308, USA
3The Findings Group, Decatur, GA 30030, USA
4Computer & Information Science & Engineering, University of Florida, Gainesville, FL 32611, USA
{erinjktruesdell, jsmith775, smathew64}@gatech.edu

Abstract

Intelligent learning environments have demonstrated effectiveness for providing individualized instruction to students of computer science (CS). However, the great potential of intelligent agents has not yet been explored within expressive environments, which are increasingly common for supporting and motivating K-12 students. This paper presents the prototype design and implementation of a novel Co-creative Artificial Intelligence (CAI) integrated within EarSketch, an online environment for learning introductory computing concepts through code-driven, sample-based music remiking. CAI is intended to scaffold student learning from EarSketch’s expressive computing curriculum by co-creating algorithmic music alongside a human learner. This paper presents an initial version of CAI, which engages with EarSketch users by offering menu-based dialogue and suggestions based on the state of a project. We report a pilot study in classrooms, showing promising results in students’ satisfaction with the system’s capabilities. The findings of this pilot study suggest the ability of a co-creative agent to support users in learning and creative objectives, and should inspire research into combined computational and creative user models.

1 Introduction

Many computer science education environments for K-12 students focus on creative expression as a means of increasing student engagement in programming (Resnick et al. 2009; Grover, Basu, and Schank 2018). This shift raises the unique challenge of offering adaptive support to learners working on open-ended creative tasks. To achieve educational objectives within creative platforms, the field needs to move toward intelligent learning support within creative environments.

Recent years have seen computational creativity research oriented towards systems that can collaborate with humans (Davis et al. 2019; Cheatley et al. 2020; Zacharakis et al. 2021; Guzdial et al. 2019).

Systems such as these are typically designed to enhance the user’s experience and produce valuable artifacts, but have not been designed to support learning. EarSketch, a platform that uses the expressive medium of sample-based music creation to engage students with programming (Magerko et al. 2016), provides an ideal environment to investigate a co-creative agent for education.

EarSketch is an expressive programming environment in which students create sample-based music (primarily in the hip-hop and electronic genres). The EarSketch interface is shown in Figure 1. EarSketch users write Python or JavaScript code to place professionally-produced sound samples and audio effects on a multi-track timeline. The accompanying curriculum covers topics relevant to programming and algorithmic music production.

EarSketch is designed for students with little to no prior experience in making music. Consequently, music theory and reading skills are not required for learners to create songs on the platform. Previous EarSketch studies have indicated strong gains in student attitudes and intent to persist in computing, especially in populations that are underrepresented in the field of computer science (Magerko et al. 2016; Engelman et al. 2017; McKlin et al. 2018).

CAI (Co-creative Artificial Intelligence) is a learning companion aimed at scaffolding learning by co-creating algorithmic music alongside a human student. Our aim with the development of CAI is to create an agent that supports the student the way a slightly more knowledgeable peer would. The current version of CAI offers this support by suggesting additions to students’ code and music, and through additional support of students’ sound selection and debugging activities in EarSketch. As students interact with CAI, the agent offers suggestions to further their creative goals for their EarSketch projects while promoting the use of more advanced coding and musical techniques. The agent supports beginner EarSketch users as well as those who have experience with the platform. CAI interacts with the EarSketch user in a non-intrusive manner, providing chat-based suggestions and assistance as students work on their
EarSketch projects. It is our goal that students who work with CAI demonstrate growth in core computing and musical learning objectives, and indicate in self-reported measures that CAI provides valuable input as they develop both aspects of their EarSketch projects.

EarSketch users select dialogue options from a menu-based interface to interact with CAI. Rather than adding code or sounds into a project, CAI offers suggestions via chat-based dialogue. This gives greater control of the project to the student and encourages them to develop projects themselves rather than relying on the agent to do so. CAI’s suggestions are made using analysis tools we developed to inform the agent about the code complexity and musical structure of the project (Smith et al. 2020). CAI presents suggestions related to advancing the project’s code and music, and aids students in sound sample selection and debugging.

We deployed a version of CAI within a pilot study in high school classrooms during the winter of 2021. Students interacted with the system, and we collected data on their interactions as well as survey data about their perceptions of CAI. The results of the survey indicate that students found CAI’s suggestions valuable and that students who interacted with more of the CAI system perceived CAI as having helped them make a better program and a better song. The remainder of this paper discusses a survey of related work; a description of CAI’s project analysis capabilities, co-creative move generation, and dialogue tree; results of a pilot study of CAI; and future areas for development and study. The results of this work suggest that co-creative AI is a promising means of applying computational co-creativity to support learners.

2 Related Work

Recent work in co-creative AI and learning companions demonstrate ongoing interest in the development of both computational creative partners as well as agents to support learning across a variety of subject areas. Co-creative systems in the domains of drawing, design, and music display multiple levels of initiative (Davis et al. 2019; Cheatley et al. 2020; Zacharakis et al. 2021; Guzdial et al. 2019). Their authors distinguish which audiences may find which levels of initiative most useful in the creative process. While intelligent learning environments have been developed for creative domains such as music composition and performance, it is not typical for these systems to be co-creative or for a co-creative system to be a pedagogical tool. CAI aims to bridge this gap by presenting pedagogy in an explicitly co-creative context, engaging EarSketch students to build competency in both musical composition and programming. This aligns with EarSketch’s goal to build confidence and interest in computing by offering an authentic expressive computing environment (Engelman et al. 2017).

2.1 Human-Computer Co-Creativity

Recent publications in computational creativity have described systems that work with humans on artistic pursuits including sketching (Davis et al. 2019), songwriting (Cheatley et al. 2020), music harmonization (Zacharakis et al. 2021), and game level design (Guzdial et al. 2019). Some of these systems, including the ALYSIA songwriting system (Cheatley et al. 2020) and the CHAMELEON harmonization tool (Zacharakis et al. 2021), were especially useful for supporting domain novices through the creative process. This dynamic is of interest for the development of CAI, which is intended to support students as they develop mastery in the EarSketch environment. Systems aimed at a broader range of mastery levels, such as Morai Maker (Guzdial et al. 2019), offer unexpected ideas and make moves in line with users’ demonstrated design style. Results of these projects have indicated that more advanced users value co-creative systems that serve in an adaptive “follower” role, while novice users typically benefit from systems that provide support using levels of knowledge which the users themselves do not have. In musical applications, co-creative artificial intelligence agents are widely used in algorithmic music composition (Lopez-Rincon, Starostenko, and Ayala-San Martín 2018). In live settings, AIs can function as “tools” for musicians or as “actors” that perform collaboratively with them (Caramiaux and Donnarumma 2020). CAI reflects these paradigms of human-AI musical collaboration by presenting users with suggestions for compositional tools such as sound samples and by evaluating a project as a student iterates on it in order to provide collaborative feedback.

Co-creative AI projects typically aim to create a satisfying interaction for the user and/or a valuable end product. Algorithm selection, interaction design, and initiative mechanisms are selected to maximize systems’ performance towards these goals. CAI extends the work in this domain: in addition to providing an enjoyable user experience and supporting the collaborative creation of valuable artifacts, CAI includes pedagogical strategies as part of the interaction between system and user.

2.2 Intelligent Learning Environments

The field of intelligent tutoring systems (ITS) was inspired by the goal of providing individualized instruction to learners (Sleeman and Brown 1982). From that line of research, learning companions arose: their goal is to support students
on their learning trajectories while leveraging the benefits of social context. Both ITS and learning companions provide personalized instruction and feedback for students. An ITS does so in a tutor or authority figure manner, while learning companions are more often modeled as peers. The architecture of these systems often includes four components: domain expertise, a model of the student, a model of a tutor or learning companion, and an interface. The student model considers how users learn and make mistakes, and the tutor model contains intervention strategies to use. This brief literature review discusses ITS and learning companion systems that support computer science and music, as the present work lies at the intersection of these two domains. All systems discussed here are designed for novice learners in that domain, usually for an introductory course on the subject.

Computer Science Intelligent Learning Environments

Intelligent learning environments for computer science generally support learning two kinds of knowledge: a particular programming language’s syntax or general computer science concepts. One of the earliest ITSs for programming is the LISP tutor (Anderson and Skwarecki 1986), which offers feedback and questions for the student specific to that programming language. Since then, many ITSs for CS have been built and investigated (Crow, Luxton-Reilly, and Wunschke 2018; Nesbit et al. 2014). CS has also seen learning companion research: learning companions have been built that support online education for Scratch (Ocaña et al. 2020) and Java (Faraco, Rosatelli, and Gauthier 2004), a robot offers feedback for tasks in the LEGO Mindstorms environment (Ahmed, Lubold, and Walker 2018), and one companion both teaches and learns about algorithms (Petry and Rosatelli 2006). These systems are task-based and do not afford open-ended or artistic coding projects. Like these systems, CAI supports learners in coding in a specific programming language (Python or JavaScript), but unlike prior systems, CAI is designed to act as a co-creative companion, supporting both aesthetic and technical decisions.

Intelligent Learning Environments for Music

Intelligent learning systems for music have been developed to focus on topics such as music theory, harmony, or playing instruments. Computer-assisted musical instrument tutoring systems (Percival, Wang, and Tzanetakis 2007) come in two varieties: specific-goal oriented projects (such as learning chords), and general-instruction systems. Piano Tutor (Dannenberg et al. 1990) and pianoFORTE (Smoliar, Waterworth, and Kellogg 1995) are two systems that specialize in teaching piano to beginner students. Piano Tutor’s expertise is in reading sheet music. The system interrogates if the student makes mistakes while practicing a piece. pianoFORTE’s focus is more advanced. It visualizes how keys are pressed, or more generally, how a piano should be performed.

More recent social learning systems have been companions whose expertise is on how instruments should be physically played. Pianobot (Ritschel, Seiderer, and Andre 2020) and instruMentor (Bagga et al. 2019) are robotic tutors that react and offer advice on piano musical performance and recorder physical performance, respectively. XR (extended reality) systems can track how students play the piano by overlaying virtual notes and feedback over keys. These include piARno (Rigby, Wünsche, and Shaw 2020), and Mixed Reality Piano Tutor (Molloy, Huang, and Wünsche 2019). The MRLS learning companion system teaches users about rhythm and allows users to create music collaboratively with one another (Wang and Lai 2011). While these systems can detect how instruments are physically played, and offer lesson plans, they do not support introducing these concepts for student-created music projects.

CAI’s scaffolding and student modeling processes are guided by practices from ITS design: the system evaluates coding concepts such as loops and modularity, and music concepts such as repetition and contrast. From other learning companions, we integrate social learning aspects and regular feedback for the changes students make to their projects. CAI incorporates these principles into its co-creative actions. By integrating modeling of students’ code and music within a co-creative system, CAI enables novice and experienced students to build domain knowledge while providing them with tools to create personally meaningful and culturally relevant artifacts (Magerko et al. 2016).

3 System Description

3.1 Analysis Module

CAI includes an Analysis Module that comprises a suite of code and music analysis tools. The Analysis Module provides CAI with snapshots of a student’s project that update while the student works and are used to help CAI select suggestions to make. These snapshots provide the system with scores describing the complexity of the student’s code, as well as a model of the musical structures and sound characteristics in the music produced by the student (Smith et al. 2020).

Code Complexity

CAI’s Code Complexity Calculator evaluates the levels at which students use 15 concepts covered in the EarSketch curriculum. The concepts are: integers, floating-point numbers, strings, Boolean values, lists, variables, mathematical operators, string operators, list operators, comparisons, Boolean logic, user-defined functions, console input, for loops, and conditional statements. Other works have adapted Bloom’s Taxonomy for the purposes of evaluating students in computer science (Starr, Manaris, and Stalvey 2008; Thompson et al. 2008). Similarly, CAI uses a knowledge taxonomy that draws from Krathwohl and Anderson’s Flattened Bloom’s Taxonomy (2009) and is tailored specifically to EarSketch learning targets. Our use of this taxonomy as a basis allows the system to model demonstrated understanding of each curriculum topic using distinct hierarchical levels.

CAI’s taxonomy enumerates 3-5 levels of complexity for each listed curriculum topic. The first complexity level for any concept is usage of it in the student’s code. The second level is usage that is not identical to sample code from the EarSketch curriculum. Subsequent levels for a concept are specific to each and outline increasingly complex uses. Table 1 describes the complexity levels for the “String” concept as an example. When the student runs a project, the
Complexity Calculator generates an object that includes representations of the student’s demonstrated complexity for each topic. The system also compares this output with the complexity object from the last time the student ran their code. The agent uses this information to select appropriate suggestions to present.

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Does not use a string</td>
</tr>
<tr>
<td>1</td>
<td>Uses a string copied directly from sample code</td>
</tr>
<tr>
<td>2</td>
<td>Uses a string not copied directly from sample code</td>
</tr>
<tr>
<td>3</td>
<td>Uses a string not copied from sample code as a function argument or in a comparison</td>
</tr>
<tr>
<td>4</td>
<td>Iterates over or indexes from a string not copied from sample code in a for loop, function argument, comparison, etc.</td>
</tr>
</tbody>
</table>

Table 1: Knowledge levels for the “String” concept.

Music Analysis The CAI Analysis Module also includes tools to analyze the musical output of a student’s EarSketch project (Smith et al. 2020). Students use EarSketch to create algorithmic music by selecting a combination of sounds and audio effects. CAI’s analysis tools combine representations of a student’s code and musical choices to represent the project as a whole. We have designed CAI to include discussion of the creative elements of EarSketch, which have led to increased positive attitudes towards computing, especially in underrepresented populations (Magerko et al. 2016). As such, the music analysis is used to gather information about students’ preferences and musical choices so that CAI can make suggestions, rather than measuring student progress in discrete levels as with the code complexity rubric.

The CAI Analysis Module creates a hierarchical representation of a student’s project in order to represent and understand its musical form or structure. Music in an EarSketch project is represented symbolically instead of as raw audio. As a script runs, sounds and effects used in the project are listed by track number to be displayed on the Digital Audio Workstation (DAW) view. The Analysis Module converts this symbolic representation from a track-based listing to a timeline view. It compares the timeline to a series of measure-to-measure distance thresholds to identify transitions between sections (large changes in instrumentation or effects that divide a song into distinct parts) and subsections (smaller changes that indicate transitional phases of a musical section). The representation of musical form created by the Analysis Module is stored in a dictionary called the Sound Profile, in which each section contains a nested structure of subsections. Every section and subsection includes the sounds and effects used at each measure within it, the parameter values for those effects, and which lines of code were used to add them to the output.

In order to identify how a student is using coding concepts to enact specific musical ideas, CAI must identify items in a student’s code and music that are used in combination. To do this, the Analysis Module allows CAI to index the Sound Profile by specifying an input and output type from the choices of section, measure, line, sound, and effect, along with an input value. The Sound Profile indexing function returns the corresponding values for anything of the specified output type that coincides with the input. For example, selecting input:measure, output:sound, input-Value:14 will return all sounds that play during measure 14 of the project. Selecting input:section, output:line, input-Value:“A” will return all lines of code that manipulate the sounds and effects in section A. This system allows CAI to observe music and code in tandem, in order to suggest actionable code changes that affect the musical output and further student understanding of both domains.

3.2 Co-Creative Moves

We observed the conversational flow between student collaborators to discover the dialogue patterns that emerged and to gain insight about creative collaboration in EarSketch to inform the development of CAI. We performed a pilot study in which we logged student-to-student communication through a chat application within EarSketch as they completed a co-creative learning task (Griffith et al. 2021).

Based upon our observations, and on our intention to make CAI an agent that promotes learning in the computational domain, we selected three primary co-creative moves as the initial set available for CAI’s interaction with students: code and music concept suggestions, sound sample recommendations, and debugging assistance. These moves were selected because they represent core parts of the experience in creating EarSketch projects: selecting samples for use in a piece of music, using code to place and manipulate those samples, and debugging code when errors arise. Furthermore, we identified these moves as key areas in which CAI aims to support students.

Other co-creative moves and dialogue tags that were found in the student-to-student studies were socially-oriented. While these are important in rapport-building and human conversation, our priority for the first version of CAI was to include the pedagogical tools necessary for the system to function. Sample selection and code suggestions work in tandem to scaffold students’ learning in both areas represented in the EarSketch platform. Debugging assistance from CAI provides context for errors and is intended to reduce student frustration. We anticipate developing additional moves for CAI in future iterations of the system.

Sound Sample Selection CAI’s ability to suggest sounds based on a student’s project supports student expression in a primary aspect of EarSketch project creation. Messages between students from our study of student-to-student co-creative dialogue included a number of exchanges where users discussed the aesthetic properties of their projects. Suggestions for changes to a project’s samples were also found in observation of collaborative live coding using EarSketch (Xambó et al. 2018). Our inclusion of sample-focused interactions supports CAI in providing an experience similar to that of collaborating with a partner, and seeks to enhance students’ experience in the expressive environment that EarSketch provides.
Additionally, CAI’s recommendation system can help students navigate the EarSketch sample library. The library includes nearly 4000 professionally-produced sounds spanning multiple popular genres of music. It includes samples from sound engineer Young Guru and EDM artist Richard Devine, as well as samples from songs by popular artists such as Ciara, Common, and Pharrell Williams. While such size benefits EarSketch users by providing a variety of sound options, the size of the library can become overwhelming to users (Smith et al. b). CAI aims to encourage students to experiment with new and potentially unfamiliar or unexpected sounds in their projects.

CAI uses its Sound Profile (see section 3.1) to create sound recommendations for specific sections of a song through an adaptation of EarSketch’s existing hybrid recommendation system (Smith et al. a). CAI’s sample recommendation nodes present 1-3 samples from the recommendation engine. When these dialogue nodes are presented to the student, recommendations are generated based upon the Sound Profile created by the Analysis Module from the most recently run version of the student’s project (and whether or not the student requested recommendations for a particular section). The sample recommendations are included in CAI’s message to the student.

### Code And Music Concept Suggestions

A primary goal of CAI is the development of students’ programming skills. One of the chief capabilities of the current version of CAI is the proposal of additions to students’ projects. The CAI agent includes a mechanism for selecting and presenting one of several dozen authored suggestions based upon the state of the user’s project. Included in the suggestion selection mechanism is a decision tree for determining which suggestion is presented, based on characteristics of the project and on what, if anything, has changed in the project’s code complexity since the student last ran their code.

Several suggestion options are targeted specifically towards changes in the student’s code complexity score. For example, if the student’s complexity score for User-Defined Functions goes from 0, “Does Not Use,” to 2, “Uses Uniquely,” between two snapshots, CAI will prompt the student to call the function they have just made. Many suggestions include additional messages for explanations and example code, which may be accessed by the student through menu options after the suggestion has been made.

### Debugging Assistance

We included debugging assistance as a core capability for CAI. While CAI offers more help with error debugging than a student partner might, we elected to include this functionality to reduce student frustration and provide additional support as users work through problems common to beginner programmers. The system stores a dictionary of all possible error types in EarSketch and a corresponding explanation of the error for each. CAI presents the explanation when prompted for error help by the student. Post-surveys of study participants indicated that students valued CAI’s assistance with error debugging.

#### 3.3 Dialogue with Student

CAI interacts with EarSketch users through a menu-based chat system (see Figure 2). The chat system contains a dialogue tree that outlines the utterances CAI can make, as well as follow-up dialogue options to present to the student. A section of the CAI dialogue tree is depicted in Figure 3. Many of the nodes in the dialogue tree contain strings that signal the system to perform actions in addition to sending an output utterance to the student. For example, [SUGGESTION] shows in several nodes in Figure 3, signals to the system that that text should be replaced by a sound recommendation for the project. Similarly, [SUGGESTION] indicates that the text in the node should be replaced by a suggestion provided by the suggestion selection mechanism. This flexibility in dialogue allows CAI to respond to a wide variety of project states while reducing the burden of dialogue node authorship.

The pane in which students chat with CAI is presented on the right-hand side of the browser window (See Figure 2). The interface is integrated into the EarSketch site, allowing CAI to respond to changes in the Code Editor in real time. The interface replaces the curriculum panel when active, and users can toggle freely between the two. Students interact with CAI by selecting dialogue options from a menu presented in the CAI interface. Menu options allow users to prompt CAI for suggestions or ask for help debugging code. CAI responds to the user in the same interface, and menu options update with each interaction as the user converses with the agent. This user interface allows CAI to continuously present information without distracting from the user’s view of the Code Editor, and the user is able to reinforce CAI’s suggestions by switching between CAI and the curriculum.

Figure 2: CAI chat interface. Users can communicate with CAI by selecting dialogue options (bottom).

### Dialogue Tree

In our initial classroom studies, we collected 3402 textual utterances from 68 pairs of students co-creating in EarSketch. We observed that, within the context of co-creative interactions between human collaborators, collaborators often conversed about coding challenges, sound choices, or both. The dialogue exchanges reflected both social and task-based interactions such as greetings, brainstorming about sound choice, sharing ideas, establish-
Figure 3: An excerpt from the CAI Dialogue Tree. In these nodes, the student is able to query CAI for sound recommendations for the project.

The interaction trajectories we observed in our foundational studies informed the design of CAI’s dialogue tree. Our goal is to emulate the patterns and elements of dialogue exchanges between human collaborators to portray CAI as a partner (André and Pelachaud 2010). To capture the social and task-based exchanges prevalent in student-to-student conversations, CAI begins with a greeting introducing itself as a “co-creative agent” that will be a partner to the student. We chose to include “I’m still learning programming” in CAI’s introduction to reinforce the idea that it is a partner and not an authoritative helper agent. CAI continues the dialogue by working with the student to select sounds for the project. If the student faces an error in the code at any point during the session, CAI’s dialogue buttons populate to provide the students with the option to ask CAI for help.

Once this initial interaction is complete, the student has multiple options for continuing their conversation with CAI as they continue to develop their projects. The student can select one of three dialogue options to re-initiate interaction, or CAI can re-initiate dialogue by presenting a suggestion when the student runs their code. If CAI has a relevant code suggestion from the code suggestion module when the student runs their code, or if the student prompts CAI for a general suggestion, CAI presents a recommendation to the student for a code and/or musical addition to the project. CAI then offers a menu option where students can request explanations for its suggestions and examples of how to use relevant code concepts, as a slightly more knowledgeable partner would, without providing the exact solution. These suggestions are based on the student model, which tracks the student’s knowledge level, and suggestion selection system. A menu item where students can query CAI for sound sample ideas for a section or the whole song is also available, and CAI’s menu populates with an error query option every time the student runs code with an error.

4 Classroom Study

We conducted a pilot study of the CAI system in three public and charter high school classrooms in districts in the South-eastern United States. The goals of this study were to determine how well the existing CAI system allows students to improve their music and code and to identify areas for improvement. Overall, 56 students consented to participate in the study, and we have pre-survey, post-survey, and project data from 42 students. The race/ethnicity makeup of the students is: White = 55%, Asian = 18%, Black = 18%, Hispanic/Latino = 7%. Further, 82% are male and 18% are female, and they range in age from 14 to 17 with a majority (64%) in 9th grade, 20% in 10th, 14% in 11th, and 2% in 12th. Students had all worked with EarSketch as part of their class curricula before the study took place. Due to the COVID-19 pandemic, students accessed CAI from both in-person classroom environments and virtual learning environments. During the class period used for the study, students were instructed to complete or continue a task in EarSketch assigned by their teacher. Students accessed the version of EarSketch that included CAI by using a special version of the EarSketch URL. Students who participated in the study completed pre- and post-surveys about their experience with coding and music and their perceptions of CAI. Data was collected about students’ interactions with CAI during the session using a series of data collection tools added to CAI to aid in system evaluation and supply interaction context for the pre- and post-survey results.

4.1 Instruments

We adapted CAI’s student modeling tools to track students’ interactions with CAI during our pilot studies. The version of CAI used for the study stores history for each student and each project directly to the EarSketch database. Tracked data include the dialogue and suggestion nodes visited by the student, as well as when the student ran the script (along with whether or not the script ran successfully, and a code complexity score if the code did run), and visits to pages in the EarSketch curriculum.

The system also notes when the student uses sound or code suggestions from the CAI agent. Code suggestion use is measured by comparing the project’s code complexity score with the expected complexity score if the student were to implement CAI’s suggestion. If the two scores match, the system considers the suggestion to have been implemented. Using this stored data, we utilized a variety of metrics to explore student-to-CAI dialogues and investigate their effects on survey outcomes. In addition, we prepared a summary for each student that included information on which nodes the student visited in which order, and information on how many of CAI’s suggestions the student implemented.

This data was collected in tandem with pre- and post-survey instruments designed to collect information on students’ perceptions of CAI. Our post-survey instrument includes questions on CAI’s cognitive support, interaction quality, concentration/flow, psychosocial support, and overall satisfaction with the system.

Questions included four-point Likert-style ratings about various aspects of the CAI system, including the agent’s
technical competency and its timing of suggestions. We use a four-point scale to encourage participants to reflect and determine whether they agree or disagree with the prompt.

In addition to the Likert-style questions, the post-survey includes free-response questions for students about why they responded they would like to work with CAI again (if they responded that they would) and about the most valuable suggestion CAI made.

4.2 Results and Discussion

Based on log file data, we compiled the number of CAI dialogue and suggestion nodes each student accessed, and correlated those numbers with student ratings of CAI on the survey items “CAI helped me make a better program” and “CAI helped me make a better song.” Students that visited a higher percentage of the nodes, and thus viewed a greater portion of the dialogue tree, were significantly more likely to agree or strongly agree with the two statements. Of the participants, 42 responded to both items, each with a mean of 2.86 on a 4-point likert scale (1 = Strongly Disagree, 2 = Disagree, 3 = Agree, 4 = Strongly Agree). We calculated correlation between response to the survey item and the percentage of tree nodes visited using Spearman’s rho: \( r(41) = .30, p = .044 \). Figure 4 summarizes these responses. The correlation between “CAI helped me make a better song” and the percentage of suggestion nodes viewed using Spearman’s rho yielded \( r(37) = .37, p = .024 \). Additionally, we analyzed open-ended survey responses from students participating in this study. As we discuss below, the results indicate that CAI’s core co-creative moves were valued by student co-creators. Students’ responses frequently mentioned CAI’s suggestion of sounds and forms, coding support, and debugging.

Sound Suggestions - Survey Responses

Students made an average of 2.4 sound sample requests per project and 1.1 code/musical structure requests per project. This indicates that students are seeking sound support more frequently than code support in the current version. Frequency of sound requests may be partially caused by students having no way to ask CAI for samples in particular genres or using specific instruments, and instead resorting to making multiple requests until CAI suggested a sample for the desired characteristic. Future versions of CAI will implement options for students to ask for sample suggestions within a specific genre. The lack of frequency of code/music structure requests may be a result of the limited number of code/music structure suggestions made available to the learner. Future iterations on the CAI suggestion system will offer a larger number and variety of suggestions. Additionally, future studies with CAI may investigate the relationship between students’ request frequencies and their previous experience with coding and making music.

The following looks at students’ open-ended responses for both sound suggestions and coding support. In response to the question, “What was the most valuable suggestion that CAI made?”, students offered that CAI’s sound suggestions were helpful: “It suggested a very good starting rhythm/sound, then suggested sounds that worked well with it” and “The beats/sounds it suggested were great.” One participant noted CAI’s help in traversing the large sound library: “[CAI was] giving me good sounds to add to my music so I don’t have to scroll through thousands of sounds to find the right one with my indecisive self.” Another student appreciated that CAI was not only making general sound suggestions, but also suggestions tailored to specific parts of the student’s project: “it suggested I use specific sounds in specific measures.”

Code Suggestions and Debugging Assistance - Survey Responses

While students made fewer code requests of CAI, many of the students found the coding support helpful. Students reacted to the prompt “What was the most valuable suggestion that CAI made?” with a number of code-related support items: “It helped me fix code” and “How to do a function.” We also asked students, “Why would you like to work with CAI again?” Students specifically commented on CAI’s support with debugging: “I would like to work with CAI because it helped me pick out sounds to add to my music and debug my program” and “Because he helped me when I had an error.”

Taken together, the results of the pilot study indicate that the initial co-creative moves implemented in CAI were successful and that CAI helped students with their EarSketch projects. Students appreciated both the aesthetic and technical moves that CAI provided. The current pilot study has several limitations. First, we did not measure student learning (for example, with a pre/post test). This study design featured self-report data commensurate with usability studies, but future studies should also measure learning. Additionally, the present study was conducted in only three classrooms in the southeastern United States, and it will be important to pilot future versions of CAI with a broader and more diverse set of users.

The combination of open-ended survey responses along with the significant, positive correlation between the number of nodes visited and student ratings of CAI suggest that students perceive CAI to be beneficial. However, the results do not establish whether more interaction with the decision tree causes greater agreement that CAI helped make a better program. It could be that a positive attitude toward CAI contributes to greater interaction with the decision tree. We will investigate this relationship in future work that analyzes whether students continue to show a positive correlation in new iterations of CAI, if the magnitude of that correlation increases, and which students tend to show positive correlations (by computing or music-making confidence, gender, or race/ethnicity).

5 Conclusion and Future Work

This paper has presented the first co-creative intelligent agent designed to support CS learning in the context of computational music remixing. We have presented CAI’s modeling tools, which analyze the musical structure and code complexity of student projects, and we have described the

---

1 Though we introduce CAI as a non-gendered agent, some students referred to CAI as “he” or “she.”
menu-based dialogues CAI engages in with users. The results of the pilot study suggest that high school students valued CAI’s suggestions, and that there was a positive correlation between students who interacted with more of CAI’s suggestions and dialogue tree and those who felt that the agent helped them to improve their song and their project, respectively. These results suggest that co-creative moves from an intelligent agent supporting CS learning objectives have the potential to improve student attitudes in both CS and a creative domain.

The findings presented here point to several important directions for future work. With regard to CAI, we will expand its functionality to include a project model that tracks goals for EarSketch projects, and augments CAI’s dialogue capabilities accordingly. More broadly, the results point to the great promise of co-creative intelligent agents for supporting learning. Future research should examine co-creativity in other educational contexts and domains, including within human-human and human-computer partnerships. Future work should also examine phenomena including turn-taking and the effectiveness of various co-creative strategies. By moving toward intelligent agents that can co-create with learners, we can provide highly effective, adaptive instruction within expressive domains.

6 Acknowledgments

This material is based upon work supported by the National Science Foundation Award No. 1814083. 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. EarSketch is available online at https://earsketch.gatech.edu.

References


Faraco, R. A.; Rosatelli, M.; and Gauthier, F. O. 2004. Adaptivity in a learning companion system. IEEE Interna-


Xambó, A.; Roma, G.; Shah, P.; Tsuchiya, T.; Freeman, J.; and Magerko, B. 2018. turn-taking and online chatting in remote and co-located collaborative music live coding.

LyricJam: A system for generating lyrics for live instrumental music

Olga Vechtomova, Gaurav Sahu, Dhruv Kumar
University of Waterloo
{ovechtom, gsahu, d35kumar}@uwaterloo.ca

Abstract
We describe a real-time system that receives a live audio stream from a jam session and generates lyric lines that are congruent with the live music being played. Two novel approaches are proposed to align the learned latent spaces of audio and text representations that allow the system to generate novel lyric lines matching live instrumental music. One approach is based on adversarial alignment of latent representations of audio and lyrics, while the other approach learns to transfer the topology from the music latent space to the lyric latent space. A user study with music artists using the system showed that the system was useful not only in lyric composition, but also encouraged the artists to improvise and find new musical expressions. Another user study demonstrated that users preferred the lines generated using the proposed methods to the lines generated by a baseline model.

Introduction
Music artists have different approaches to writing song lyrics. Some write lyrics first, and then compose music, others start with the music and let the mood and the emotions that emerge from the music guide their choice of lyric themes, words and phrases. The latter type of songwriting is often experimental in nature, and is commonly found in genres such as rock, jazz and electronic music, whereby one or more artists jam and in the process converge on the desired musical expressions and sounds. In this work, we describe a system we designed to support this type of song writing process, where artists use musical instruments, while the system listens to the audio stream and generates new lyric lines that are congruent with the music being played in real time. The artists see the lines as they are generated in real time, potentially entering a feedback loop, where the lines suggest phrases and themes that artists can use to guide their musical expressions and instrumentation as they play their instrument. The generated lines are not intended to be the complete song lyrics, instead acting as snippets of ideas and expressions inspiring the artist’s own creativity. After the jam session is over, the lines are saved, and the artist can use them as inspiration to write the lyrics for their song.

Our intent is to design a system capable of generating original lyric lines that match the emotions and moods evoked by live instrumental music. Past research in musicology has found a correlation between some aspects of music and emotions. In one large-scale study, researchers found evidence that certain harmonies have strong associations with specific emotions, for example, the diminished seventh is associated with the feeling of despair (Willimek and Willimek 2014). In another study, researchers found correlation between major chords and lyric words with positive valence (Kolchinsky et al. 2017). In addition to harmonies, various sound textures and effects also contribute to the emotional intent of the music. Therefore, in this work we use raw audio input that captures all aspects of music.

The proposed model is trained in an unsupervised manner, being only shown aligned data, consisting of an audio clip and its corresponding lyric text. No labels have been assigned to the data. During training, the model can learn any discernible associations between the raw audio characteristics captured in the spectrograms and the texts of lyrics. At inference time, the model receives clips sampled from the live audio stream of the music played by the user, and generates new lines. The associations learned by the model determine various characteristics of the generated lines, which could include sentiment and stylistic markers, lexical expressions, syntactic characteristics and syllable patterns.

Our approach is based on training a variational autoencoder for learning the representations of Mel-spectrograms of audio clips (spec-VAE), and a conditional variational autoencoder for learning the representations of lyric lines (text-CVAE). The advantage of using variational autoencoders as generative models is their ability to learn a continuous latent space that can then be sampled to generate novel lines, which is an important requirement for creative applications.

At inference time, the model must be able to generate new lyric lines given an audio clip being recorded from live jam session. In order to do that we need a way to align the latent representations learned by the spec-VAE and the latent representations learned by the text-VAE. We propose two novel approaches to achieve this alignment.

The first approach (Figure 1) is based on training a separate Generative Adversarial Network (GAN) model that takes the spectrogram embedding from spec-VAE, and learns to predict the lyric line embedding in the text-CVAE. The GAN-predicted embedding is then sent to the text-CVAE decoder to generate text.
The second approach (Figure 2) learns to transfer the latent space topology of the spec-VAE to the text-CVAE latent space. To achieve this we use the learned posteriors from the spec-VAE as priors in text-CVAE during its training. The text-CVAE learns to encode lyric lines corresponding to a given audio clip in the region of the latent space corresponding to that clip. Also, since similar sounding audio clips are encoded in neighboring regions, the text-CVAE correspondingly learns to encode lines for these clips in neighboring regions. For example, ambient music clips would be encoded in neighboring regions of spec-VAE, and so would be the lines corresponding to these clips. The intuition is that lines corresponding to similar sounding audio clips (e.g. ambient) would have similar emotional intent, as opposed to, for example, aggressive sounding music. At inference time, when an artist plays an ambient music piece, the system would feed its spectrogram to the spec-VAE encoder to get the parameters of its posterior distribution. Since the spec-VAE posterior distributions are also prior distributions in the text-CVAE, the system samples latent codes from the corresponding prior of the text-CVAE, generating new lines reflecting ambient music.

To summarize, the contributions of this paper are as follows:

1. To our knowledge this is the first real-time system that receives a live audio stream from a jam session and generates lyric lines that are congruent with the live music being played.
2. We propose two novel approaches to align the latent spaces of audio and text representations that allow the system to generate novel lyric lines matching the live music audio clip.
3. We discuss our findings based on observations and interviews of musicians using the system.

In the next section, we describe the GAN-based approach of aligning the spec-VAE and text-CVAE latent embeddings. Next, we will present our second approach of aligning the topologies of the two VAEs. We will then describe user studies and present our findings, which will be followed by the discussion of related work, implementation details and conclusions.

**Approach 1: GAN-based alignment of music and lyrics representations**

The model consists of three neural network models that are trained consecutively in three stages (Figure 1).

**Training stage 1: spec-VAE**

In this stage, we train the spectrogram variational autoencoder (VAE) model to learn the latent representations of audio clips.

First we convert the raw waveform audio files into Mel-spectrogram images using the same method as used in Vech
tomova, Sahu, and Kumar (2020). These spectrograms are then used as input for the spec-VAE.

The variational autoencoder (Kingma and Welling 2014) is a stochastic neural generative model that consists of an encoder-decoder architecture. The encoder transforms the input image $x$ into the approximate posterior distribution $q_\phi(z|x)$ learned by optimizing parameters $\phi$ of the encoder. The decoder reconstructs $x$ from the latent variable $z$, sampled from $q_\phi(z|x)$. In our implementation, we use convolutional layers as the encoder and a deconvolutional layers as the decoder. Standard normal distribution was used as the prior distribution $p(z)$. The VAE is trained on the loss function that combines the reconstruction loss (Mean Squared Error) and KL divergence loss that regularizes the latent space by pulling the posterior distribution to be close to the prior distribution.

**Training stage 2: text-CVAE**

Unlike the vanilla VAE used for encoding spectrograms, we use conditional VAE (CVAE) for encoding lyrics.

The CVAE learns a posterior distribution that is conditioned not only on the input data, but also on a class $c$: $q_\phi(z|x,c)$. Here, we define the class as the spectrogram corresponding to a given line. Every conditional posterior distribution is pulled towards the same prior, here the standard normal distribution.

During training, every input data point consists of a lyric line and its corresponding spectrogram. We first pass the spectrogram through the spec-VAE encoder to get the parameters of the posterior distribution (a vector of means and a vector of standard deviations). We then sample from this posterior to get a vector $z^{(s)}$ that is then concatenated with the input of the encoder and the decoder during training. The reason why we used sampling as opposed to the mean $z^{(s)}$ vector is to induce the text-VAE model to learn conditioning on continuous data, as opposed to discrete classes. This prepares it to better handle conditioning on unseen new spectrograms at inference.

Both the encoder and the decoder in the text-CVAE are Long Short Memory Networks (LSTMs). The sampled $z^{(s)}$ is concatenated with the word embedding input to every step of the encoder and the decoder.

**Training stage 3: GAN**

In this phase, we train a generative adversarial network (GAN), which learns to align audio and text latent codes. The GAN architecture has a generator $G$ and a discriminator $D$. For training the GAN on a given spectrogram-text pair $\{x^{(s)}, x^{(t)}\}$, we follow these steps:

1. First, we pass the spectrogram $x^{(s)}$ through spec-VAE to obtain $z^{(s)} = \mu^{(s)} + \tau (\epsilon \cdot \sigma^{(s)})$, the latent code sampled from the posterior distribution. Here, $\mu^{(s)}$ and $\sigma^{(s)}$ denote the mean and standard deviation predicted by the spec-VAE, $\epsilon \sim \mathcal{N}(0, 1)$ is a random normal noise, and $\tau$ is the sampling temperature. Simultaneously, we obtain $z^{(t)} = \mu^{(t)} + \tau (\epsilon \cdot \sigma^{(t)})$ by passing the corresponding lyric line $x^{(t)}$ through the text-VAE.

2. After obtaining $z^{(s)}$ and $z^{(t)}$, we proceed with the GAN training. We pass $z^{(s)}$ through the generator network, which outputs a predicted text latent code $\hat{z}^{(t)}$.

3. We then pass $\hat{z} = [\hat{z}^{(t)}; z^{(s)}]$ and $z = [z^{(t)}; z^{(s)}]$ through

\[
KL = KL(q_\phi(z^{(t)}|x^{(t)}, z^{(s)}) || p(z^{(t)}|x^{(t)}))
\]
the discriminator network, where \( \phi \) denotes the concatenation operation. We treat \( \hat{z} \) as the negative sample, and \( z \) as the positive sample. The discriminator \( D \) then tries to distinguish between the two types of inputs. This adversarial training regime, in turn, incentivizes \( G \) to match \( \hat{z}^{(t)} \) as closely as possible to \( z^{(t)} \).

The adversarial loss is formulated as follows:

\[
J_{GAN} = \min_G \max_D V(D, G) = \mathbb{E}_{x \sim D_{\text{train}}} [\log D(z) + \log (1 - D(\hat{z}))]
\]

where \( D_{\text{train}} \) is the training data, and each sample \( x = \{x^{(s)}, x^{(t)}\} \). We also add an auxiliary MSE loss to the objective function as it is found to stabilize GAN training (Khan et al. 2020). The overall loss for the GAN is:

\[
J_{GAN} = \min_G \max_D V(D, G) + \lambda_{MSE} ||\hat{z}^{(t)} - z^{(t)}||
\]

At inference time, the encoder of the text-CVAE is no longer needed. A spectrogram is input to the spec-CVAE encoder to obtain the spectrogram latent code \( z^{(s)} \), which is then fed to the generator of the GAN, which generates the lyric latent code \( z^{(t)} \). The inference process is also stochastic, as \( z^{(s)} \) is sampled from the posterior distribution for \( s \). Sampling allows us to generate diverse lines for the same spectrogram. The GAN-predicted text latent code is then concatenated with the spectrogram latent code and input to the text-CVAE decoder, which generates a lyric line.

**Approach 2: Latent space topology transfer from spec-VAE to text-CVAE**

The intuition for this approach is to induce the text-CVAE to learn the same latent space topology as the spec-VAE. This would mean that data points that are close in the spec-VAE latent space are expected to be close in the text-VAE latent space. More concretely, if two audio clips are encoded in the neighboring regions of the spec-VAE latent space, their corresponding lyric lines should also be encoded in the neighboring regions in the text-CVAE latent space.

The training is a two-stage process (Figure 2), where the first stage, spec-VAE, is the same as in the GAN-based approach. For the second stage, we train a different formulation of text-CVAE. Instead of using one prior (standard normal) to regularize every posterior distribution, we use the posterior of the spec-VAE as the prior for any given data point. More formally, let the spectrogram be \( x^{(s)} \) and its corresponding lyric line be \( x^{(t)} \). The posterior distribution for the spectrogram in the spec-VAE is \( q_\phi^{(s)}(z^{(s)}|x^{(s)}) \), and the posterior distribution for the lyric line in the text-CVAE is \( q_\phi^{(t)}(z^{(t)}|x^{(t)}, z^{(s)}) \). The KL term of the text-CVAE loss is computed between the posterior for the lyric line and the prior which is set to be the posterior of its corresponding spectrogram in spec-VAE:

\[
\text{KL}^{(t)} = \text{KL}(q_\phi^{(t)}(z^{(t)}|x^{(t)}, z^{(s)})||q_\phi^{(s)}(z^{(s)}|x^{(s)}))
\]

The cross-entropy reconstruction loss for text-CVAE is:

\[
J_{rec}(\phi, \theta, z^{(s)}, x^{(t)}) = -\sum_{i=1}^{n} \log p(x^{(t)}|z^{(t)}), \quad z^{(s)}, x_1^{(t)} \ldots x_{i-1}^{(t)}
\]

The final text-CVAE loss is the combination of the reconstruction loss and the KL term:

\[
J_{CVAE}(\phi, \theta, z^{(s)}, x^{(t)}) = J_{rec} + \lambda \text{KL}^{(t)}
\]

To avoid the KL term collapse problem, we used two techniques first proposed by Bowman et al. (2016): KL term annealing by gradually increasing \( \lambda \) and word dropout from the decoder input.

**Ranking of generated lines with BERT**

At inference time, the model generates a batch of 100 lines conditioned on a short audio clip sampled every 10 seconds from the user’s audio source. Since the system shows one or two lines to the user for each clip, and since not all generated lines are equally fluent or interesting, we need a method to rank them so that the system shows a small number of high-quality lines to the user. For this purpose, we used a pre-trained BERT model (Devlin et al. 2019), which we fine-tuned on our custom dataset. The dataset consisted of 3600 high-quality and low-quality lines that were manually selected by one of the authors from a large number of lines output by a VAE trained on the same dataset as used in our experiments. The criteria used for determining quality of lines included the following: originality, creativity, poetic quality and language fluency. While BERT is trained as a binary classifier, we use logits from the final layer for ranking the lines.

**Live lyric generation system**

We developed a React/NodeJS web application\(^2\) that listens to the user’s audio source, which can be either a microphone or line level input from the user’s audio-to-digital converter, receiving input from the user’s instruments, such as guitar or keyboard. The application samples clips from the audio stream every 10 seconds and saves them as uncompressed PCM WAV files at 44.1kHz sampling rate. On the server, WAV files are converted to Mel-spectrograms and sent to the spec-VAE to obtain the latent code, which is then used in lyric generation by the text-CVAE. The lines generated by text-CVAE are passed through BERT classifier for ranking. The top-ranked lines are then displayed to the user on their screen. The lines slowly float for a few seconds, gradually fading away as newly generated lines appear. The user can view the history of all generated lines with time stamps during the jam session in a collapsible side drawer (Figure 3).

\(^2\)The application can be accessed at: https://lyricjam.ai
Evaluation

We trained the system on the aligned lyric-music dataset (Vechtomova, Sahu, and Kumar 2020), consisting of 18,000 WAV audio clips of original songs and their corresponding lyrics by seven music artists in the Rock genre.

The goals of our evaluation are two-fold:

1. Determine whether the two proposed methods generate lines that match the mood created by the music more accurately.

2. Understand how the application can benefit music artists as they compose new songs by playing musical instruments.

To answer these evaluation questions, we designed two user studies, described in detail in the subsequent sections.

Study 1

For this study, we developed an interactive web application, which plays instrumental songs and every 10 seconds shows three lines to the users in random order, two generated by the proposed models and one by the baseline model. The user was asked to click on the line that best matches the mood of the music currently being played. Participants for this study did not have to be songwriters or music artists, but they were required to have a general appreciation of music of any genre.

For the baseline, we used CVAE with standard normal prior. This is a strong baseline, which generates lines conditioned on the music being played. Comparison to this baseline specifically lets us evaluate the effect of the two proposed methods: GAN-based alignment (GAN-CVAE) and topology transfer (CVAE-spec). At inference time, the lines generated by each system were ranked by BERT, and the top-ranked line from each system was selected for the experiment.

We selected five instrumental songs by two artists for this study with a variety of tempos and instrumentation, and evoking different moods. The lines for each 10-second song clip were generated in advance, so that each user was shown exactly the same three lines per clip. The order of line presentation was random each time. Examples of generated lyrics and the spectrograms of corresponding audio clips can be seen in Figure 4.

In total, 15 users participated in this study. As can be seen from Table 1, users preferred the lines from two experimental systems over the baseline regardless of the type of song played. The differences are statistically significant (ANOVA, \( p < 0.01 \)).

Study 2

For this study we asked five musicians to use the system as they played musical instruments. The participants were free to choose any musical instruments they liked and play any genre of music. Below is the summary of the instruments and genres played by the participants:

- Participant A: electric guitar / rock
- Participant B: keyboard / rock
- Participant C: guitar and electric drums / blues, rockabilly, punk, folk, rock.
- Participant D: piano / new age, contemporary classical, Bollywood, pop;
- Participant E: electric guitar / metal

The participants were asked to observe the generated lines as they played music, and answer the following questions after the completion of the study:

1. What were your thoughts after using the application?
2. What did you like about it?
3. What would you change about this system?
4. Would you consider using this system again?
5. Did the lines reflect the mood that your music conveys to you?

Since this is an exploratory study aimed at forming a better understanding of how artists may use such a system, we chose open-ended questions in order not to bias the participants and let them speak about their unique experiences while using the system and ways in which this interaction may have affected their creative processes.

All users found the experience enjoyable and would use the system again. Three users mentioned that they liked its simplicity and minimalist design. Interestingly, while our initial goal was to design a system to assist musicians in lyric writing, two unexpected new uses emerged from this study: (a) the system encouraged improvisation and trying new sounds and (b) the system was perceived as a helpful jam partner. This suggests that the system could be useful not only for lyric composition, but also for music playing in general. Below, we elaborate in more detail on the themes that emerged from the feedback.

**Improvisation and experimenting with new sounds.**

Users commented that the system encouraged them to try new sounds and experiment with new musical expressions. User A: “[The system] inspires me to try things out to see what this sound does to the machine”. User C: “there were a few times where the system started suggesting lyrics that got me thinking about structuring chords a bit differently and taking a song in different directions than originally intended”.

User A also commented that the system encouraged them to be more experimental in their music: “You become much less critical of making unmusical sounds and you can explore the musical palette of the instrument to see what lines it provokes.” This user also observed a game-like element of the system: “I am trying to make it say something crazy by playing crazy sounds” and “I am playing my guitar out of tune and try to find music in it as the machine is trying to find meaning in it.” The user also noted that the generated lines sometimes encouraged them to stay in a certain music mode, such as minor chords, when they liked the lines that it generated. This suggests that the system could act as a responsive listener for musicians, to help them bring their music to the desired emotional expression. For example, if the musician intends their music to be sombre, and the emotions of the lines match the mood, they would know they are on track, and if the emotions in the lines start shifting as they try new expressions, it could be a signal that their music may not have the intended emotional effect.

<table>
<thead>
<tr>
<th>Song description</th>
<th>CVAE</th>
<th>CVAE-spec</th>
<th>GAN-CVAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthesizer, bass guitar, banjo, piano / forlorn, tense, slow</td>
<td>162</td>
<td>152</td>
<td>195</td>
</tr>
<tr>
<td>Electric guitar, synthesizer / harsh, aggressive, high-tempo</td>
<td>61</td>
<td>94</td>
<td>89</td>
</tr>
<tr>
<td>Modular synthesizer, keyboard / melancholy, low-tempo</td>
<td>64</td>
<td>48</td>
<td>76</td>
</tr>
<tr>
<td>Piano / sombre, tense, slow-tempo</td>
<td>43</td>
<td>91</td>
<td>66</td>
</tr>
<tr>
<td>Modular synthesizer, keyboard / mellow, uplifting, ambient</td>
<td>53</td>
<td>72</td>
<td>53</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>383</strong></td>
<td><strong>457</strong></td>
<td><strong>479</strong></td>
</tr>
</tbody>
</table>

Table 1: Number of lines per system and instrumental song selected by the users in Study 1

**System as a jam partner.** One user commented that the system acts like a jamming partner as they play: “I don’t feel like I am alone. I feel like jamming with someone.” “It’s like having an uncritical jam partner. It always responds to whatever you are trying.” “The machine is trying to work with you even if you make a mistake.” This suggests that a musician can benefit from using the system and be encouraged to play their instrument, even if they are not actively trying to write lyrics.

**System as a source of lyrics.** Users commented that the system would be useful as a source of new lyrical ideas. User C: “I could see it being very useful if I’m having a bit of writer’s block and just noodling around to get a starting point.” User B: “I like to jam with my band. We jam for 30 minutes or so. It would be nice to have it running. It would do the work for you as you are jamming. If at the end of it you have a pile of lyrics that would be great. That could be the starting material we could use for a song.” The user emphasized that for them it would be important to see the lines after the jam session as they may miss some lines while playing their instruments: “I look at my hands when playing, occasionally glancing at the screen to see the lyrics. I would like to see the lines after the jam session.”

**Generated lines and music mood.** Users noticed differences in lines, in particular with the changes in tempo and instruments used. One user played on their keyboard in the piano and Hammond B3 (organ) modes. The user played minor piano chords on the piano, noticing that the lines were darker in theme. Switching to organ with a more upbeat percussive sound and using major chords more often led to more upbeat lines.

Another user noticed that the system generated interesting lines when they played unusual musical phrases: “The system seemed to produce more interesting lines when it was fed slightly more novel audio, e.g., a tune that had a mode change (switching from major to minor)”. The genre of music did not appear to have as strong effect as tempo or the choice of instruments. One user who played blues, rock, folk and rockabilly genres did not observe a strong effect of genre. The likely reason is that the training dataset only contains rock music, and therefore the model has never seen music of other genres. It would be interesting to see how the lines are affected by genre, if the model is trained on a larger dataset, which is more representative of different genres.
Themes in generated lyrics

Users noted presence of recurring themes as they played. They commented on often seeing recurring words in different lines shown to them when they played a musical composition. This is somewhat expected, since most musical compositions have repeating musical phrases which prompt the system to generate lines with similar words. Also, the frequently observed words noted by the users appeared to be different for each user, which suggests that the system differentiated between their musical styles, and narrowed down on specific themes. One user commented that when they experimented by playing songs with overtly obvious emotional valence (sad and happy), the system generated lines with markedly different words: "pain", "falling", "leaving" for a sad song, and "time", "dreams", "believe" for a happy song.

Training the model on a larger dataset would likely lead to more lexical diversity. Furthermore, during the experiments, the system always showed the users top two generated lines ranked by BERT, which was fine-tuned on a small dataset as well. When we changed this setting to sampling lines from the top 10 ranked lines, the diversity improved.

Related Work

For decades, music artists and poets used various techniques to find novel forms of lyrical and poetic expressions. The Dadaist movement of the early twentieth century gave rise to the cut-up technique, as a way to use randomness to free up the unconscious from the confines of logic and tradition (Lewis 2007). The technique was popularized by William Burroughs, who viewed language ("the Word") as a lock that restricts our creativity and confines us into predictable patterns of perception and expression. He saw cut-up as a way for both the artist and the reader to free themselves from this lock. Music artists, such as David Bowie and Kurt Cobain (Jones 2015), used cut-up technique extensively in their work to inspire creativity. In 1995, David Bowie and Ty Roberts co-created Verbasizer, a system that randomizes several text inputs and produces unexpected word combinations, which David Bowie then used as inspiration for his lyrics (Braga 2016). There has also been an interest in music and research communities to design systems for live musical improvisations. Notably, Biles (1994) designed GenJam, a system that generates jazz music improvisations. It was one of the pioneering works that envisioned a system as a co-creative partner in live music performance.

Automated creative text generation has long been getting attention from researchers at the intersection of computational creativity and artificial intelligence. Creative text generation includes tasks such as poetry (Manurung 2004; Zhang and Lapata 2014; Yang et al. 2017) and story (Nair and Hinton 2010). The decoder is a mirror image of the encoder, with four ConvTranspose2d layers, interleaved with ReLU activations and one Sigmoid activation in the end. We use 128 latent dimensions for the mean and sigma vectors. During training, we use a batch size of 32, learning rate of 1e-4, and Adam optimizer (Kingma and Ba 2015). The sampling temperature is 1.0 for both training and inference.

Spec-VAE. PyTorch (Paszke et al. 2019) is used to implement the spec-VAE. The encoder has four Conv2d layers interleaved with ReLU activation function (Nair and Hinton 2010). The decoder is a mirror image of the encoder, with four ConvTranspose2d layers, interleaved with three ReLU activations and one Sigmoid activation in the end. We use 128 latent dimensions for the mean and sigma vectors. During training, we use a batch size of 32, learning rate of 1e-4, and Adam optimizer (Kingma and Ba 2015). The sampling temperature is 1.0 for both training and inference.

GAN. We use the AllenNLP library (Gardner et al. 2018) to implement the GAN. The generator and discriminator networks are 3-layered feed-forward neural networks, interleaved with ReLU activation function. During training, we use a batch size of 32, learning rate of 1e-3, and Adam optimizer for both the generator and the discriminator. We set

Implementation Details

Text-CVAE. We use the Tensorflow framework (Abadi et al. 2016) to implement the text-CVAE. The encoder is a single-layer bi-LSTM and a decoder is an LSTM. The hidden state dimensions were set to 300 and the latent space to 128. The CVAE models in our experiments were trained for 500 epochs.

Spec-VAE. PyTorch (Paszke et al. 2019) is used to implement the spec-VAE. The encoder has four Conv2d layers interleaved with ReLU activation function (Nair and Hinton 2010). The decoder is a mirror image of the encoder, with four ConvTranspose2d layers, interleaved with three ReLU activations and one Sigmoid activation in the end. We use 128 latent dimensions for the mean and sigma vectors. During training, we use a batch size of 32, learning rate of 1e-4, and Adam optimizer (Kingma and Ba 2015). The sampling temperature is 1.0 for both training and inference.

GAN. We use the AllenNLP library (Gardner et al. 2018) to implement the GAN. The generator and discriminator networks are 3-layered feed-forward neural networks, interleaved with ReLU activation function. During training, we use a batch size of 32, learning rate of 1e-3, and Adam optimizer for both the generator and the discriminator. We set
$\lambda_{\text{MSE}} = 1.0$ to ensure diverse lyric lines. The sampling temperature is 1.0 during both training and inference. The GAN alignment network was trained for six epochs.

**BERT.** We use the Transformers library (Wolf et al. 2020) to fine-tune a BERT-base model for sequence classification on our custom dataset. The model is trained for 15 epochs using a learning rate warmup scheduler for the first 500 steps of training with a weight decay of 0.01. We use a batch size of 16 for training and 64 for inference.

**Conclusions**

We developed a system for real-time generation of lyrics matching the live instrumental music being played by an artist during a jam session. Two novel approaches to align the learned representations of music and lyrics have been proposed: GAN-CVAE, which adversarially learns to predict the lyric representation from the music representation, and CVAE-spec, which transfers the topology of the music (spectrogram) latent space learned by the spectrogram VAE to the lyric latent space learned by the text CVAE. Our user study showed that users selected the lines generated by the proposed two methods significantly more often than the lines generated by a baseline for different types of instrumental songs. For another user study we recruited musicians performing live music. Their feedback suggested that the system could be useful not only for lyric writing, but also (a) to encourage musicians to improvise and try new musical expressions, and (b) act as a non-critical jam partner.

The user studies in this work as well as statistical analysis in our prior research indicate that it is possible to learn semantic and emotional associations between music audio and lyric texts in an unsupervised manner. As the follow-up to this work, we are conducting further analysis of what other associations between lyric texts and music the unsupervised neural network models are learning.

**Acknowledgments**

This research was partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant No. RGPIN-2019-04897 and Compute Canada (www.computecanada.ca).

**References**


Braga, M. 2016. The Verbasizer was David Bowie’s 1995 lyric-writing music app.


4. Social Dimensions
Walking the Line in Computational Creativity:
Paradox and Pitfall in the Mitigation of Automated Offense

Tony Veale
School of Computer Science, University College Dublin, Ireland.
Tony.Veale@UCD.ie

Abstract
Social media is now the megaphone of choice for many digital-age provocateurs. Social networks offer a wealth of examples of egregious misbehaviour by humans, but perhaps more worryingly, they also serve to magnify and weaponize the automated outputs of generative systems. So what responsibilities do the creators of these systems bear for the offenses caused by their creations, and what can they do to prevent, or mitigate, the worst excesses? For those who “make things that make things,” the issue is further complicated by the potential of our systems to accidentally tread on the feelings of others. When does a norm become a troubling stereotype, and how should a system built on social norms avoid crossing the line? We argue that because the good to the bad to the ugly.

Clockwork Lemons
If beauty is in the eye of the beholder, then so, frequently, is offense. Although a shared culture imposes many common values and norms, each of us brings our own sensitivities to bear when judging aspects of creativity and matters of taste. Polite society lays down some bold red lines, but we draw our own lines too, around the acceptable use of words and symbols, around the meanings that those signs can convey, and around the unspoken assumptions that underpin them. As social networking tools extend the reach of our digital provocations, they increase the chances that others will see them—and as a consequence us—as objectionable. The same potential for offense increasingly attaches to the outputs of our generative systems, especially when those outputs use much the same signs to evoke much the same meanings. So in what ways are these systems beholden to their beholders, and where do we draw the line so as to minimize offense?

To lean on a cinematic cliché, the outputs of a generative system run the gamut from the good to the bad to the ugly. The “good” represents the idealized targets of our system, a body of outputs exhibiting both novelty and usefulness that embody the desirable qualities of an inspiring set (Ritchie, 2007). The “bad” represents cases in which the system falls short of those ideals, with outputs that cleave too closely to the landmarks of an inspiring set—imitation and pastiche—or land too far away to be valued as meaningful or useful. While the “ugly” represents the worst of the rest, the set of outputs most likely to cross the line and cause real offense, it also overlaps as much with the “good” as with the “bad.” Ugliness is not the absence of beauty, nor is it its opposite. These are complementary qualities that can sit side by side in the same outputs. One can craft an elegant turn of phrase or a masterly image to express a truly abhorrent viewpoint. If the ugly were just a subset of the bad, we could reduce the potential for offense by simply making a system better at what it is meant to do. However, as one can sit cheek by jowl with the other, an appealing surface is no guarantee that an appalling message is not lurking beneath. Ventura and Gates (2018) propose that, for certain kinds of creative system—such as those that generate policies and plans—ethical issues can be quantified much like aesthetic issues, with objective functions that score different kinds of value. In a semiotic system of cultural signifiers, however, ethical issues are often subtle, subjective, and difficult to quantify.

At the most general level, there are two broad approaches to regulating offense: inner regulation and outer regulation. Inner regulation of a generative system curtails a state space and the rules for exploring it, so as to prevent offenses from being committed in the first place. Such a regulator aims to remove offensive capability from a system at the outset, so that it can never even imagine an ugly possibility, perhaps because it lacks the words, concepts or other tools to do so. Outer regulation sees this hope as naive, since most offense is emergent: it is not a property of the parts but of the whole. An outer regulator is an in-built censor that allows a system to generate what it will, before making ethical judgements about which outputs are safe enough to see the light of day. In Freudian terms, inner regulators curb the id of a system, while outer regulators impose a super-ego to filter this id.

To navigate these options, the paper is laid out as follows. We start by revisiting some extreme examples of offensive online content and/or behaviour, by humans and machines. Where did these creators go wrong, and what could an outer regulator have done to prevent them? We then consider the smallest possible ways in which our systems can offend, by surveying some of the simplest generative systems, or bots. Bots allow us to unpick content and behaviour, to see that it is often a combination of the two that causes offense.
regulators focus on the parts, not on the whole, as shown by our survey of dictionary-based approaches. Only by shifting to a more holistic viewpoint can we appreciate how offense is all too easily generated by systems with no obvious bias or malicious intent. We must thus recognize the potential for a system’s innocent choices to be interpreted as deliberate provocations, since audiences often place a more nuanced spin on outputs than a system itself knows how to generate.

There are many reasons why we in the community build systems whose generative reach exceeds their interpretative grasp. We do so to make the most of meagre resources, or to allow for occasions when a system surprises us much as its target audience. We do so when a system is rich in data but poor in knowledge, or just because shallow generation is easier than deep interpretation (a clever framing strategy can convince an audience of the presence of the latter; see Charnley et al., 2012). We do so when the divide between a system’s reach and its grasp impresses much more than it disappoints, as when it fosters more Eliza than Tale-Spin effects (Wardrip-Fruin, 2009). The Tale-Spin effect, named after the seminal story generation system (Meehan, 1977) shows a system mashing its gears and producing the wrong kind of sparks, while the Eliza effect, named for the famous (and infamous) chatbot, allows systems to take credit where none is deserved. As shown by Weizenbaum’s Doctor script for Eliza (1966), users often ascribe insight and intent where there is none. Yet this effect cuts both ways. If designers are willing to accept good Eliza effects, they must also accept its ugly effects too, as when insight and intent are ascribed to a system’s accidentally insensitive or offensive outputs.

No solution to offense avoidance is without pitfalls, and most breed strange paradoxes of their own. For instance, an outer regulator of vulgarity must also be an inner regulator, so as to detect and filter what should not be said. Likewise, a regulator of racist or sexist tropes must be well-versed in the ways of belittling others, so as to avoid the same traps. A regulator aiming to prevent gender-stereotyping must be subtly gender-normative, and reinforce certain norms so as to preempt any outputs that frame the atypical as abnormal. Most systems tacitly incorporate a model of their audience, after a fashion, in their objective functions, in their aesthetic criteria, and in the ways they frame their outputs, but those audience models must increasingly integrate a clear sense of what affronts as much as what delights. In fact, a robust super-ego is essential for systems that adapt to their users to grow their spaces or to evolve new objective functions, so that those systems are not corrupted over time, or worse, weaponized by their most malicious users to attack others.

In this position paper we hope to stimulate debate within the computational creativity community, by examining the responsibilities that we all bear as system builders, that our systems bear as meaning makers, and that the larger public bears as potential end-users and informed critics. The issues are much larger than any single paper or project can hope to address, but we can start by confronting the assumptions that underpin our conceptions of offense, before outlining some strategies for mitigating its most insidious forms.

Epic Fails

Even in the age of “going viral,” a single nugget of content cannot make a career, but a single tweet can still ruin one. Consider the case of comedienne Roseanne Barr who, back in 2018, led the revival of her eponymous sitcom. The new show was feted by critics, but came to a crashing halt after Barr tweeted the following in the early hours of May 29:

**muslim brotherhood & planet of the apes had a baby=vj**

The “vj” of her tweet refers to Valerie Jarrett, an appointee of the Obama administration and a women of colour. While Barr’s conceit is a technically deft conceptual blend of two very different input spaces (Fauconnier & Turner, 2002), it draws on an odious animal trope long favoured by racists. It is not enough to judge blends on technical grounds alone; unlike Barr, we (and our systems) cannot be so charmed by a novel conceit that we are blinded to its intrinsic ugliness. Barr soon regretted her tweet, but after-the-fact evaluation often comes too late, and at a high price. Barr was quickly fired by her network from a show that once bore her name.

Barr undoubtedly recognized her own use of this trope, but did not consider it ugly until her career was threatened. We need our generative systems to do both of these things at the same time: to recognize the tropes that seem to apply to their outputs, at least in the minds of an audience, and to recognize the potential for harm contained within them. A failure on each of these fronts had been the undoing in 2016 of a flagship Twitterbot by Microsoft, called @TayAndYou. “Tay,” designed as a conversational chatbot, was given the language model of a curious teenager. Although this model was carefully stocked with anodyne content, Tay was also designed to learn from those it interacted with, and to adopt stances in its tweets rather than simply sit on the fence. Tay would prove that inner regulation is a good starting point for an interactive bot, but no amount of curated content for hot-button issues – Tay had upbeat views on Caitlin Jenner (“a hero & a stunning beautiful woman”) and sympathetic views on recent victims of police violence – could stop its generative model being overwhelmed by malign influences. It was soon parroting very different takes on these topics:

**caitlin jenner pretty much put lgbt back a 100 years as he is doing to real women**

Despite Microsoft’s best intentions, Tay was a signal failure of outer regulation in a bot. Even a naïve filter would have found Tay’s repeated use of ethnic slurs and racial epithets offensive, and identified topics of great sensitivity where a bot like this should never dare to tread. Tay dared, however, and was soon denying one genocide (the Holocaust) while advocating another (of minorities in the United States). Barr compared a black Obama appointee to an ape in her tweet, but Tay would describe Obama himself as a monkey, and – in the same tweet – accuse George W. Bush of planning the 9/11 attacks. Microsoft was forced to euthanize its bot less than 24 hours of it going live on Twitter, much as the ABC television network was later moved to cancel Barr.
Microsoft blamed the bot’s rapid slide to the dark side on “a coordinated attack by a subset of people [that] exploited a vulnerability in Tay” (Ohlheiser, 2016). This vulnerability was not a secret backdoor or a code glitch, but a gaping hole in its design. Microsoft failed to provide its bot with even the most rudimentary outer regulator, relying instead on the kindness of strangers to treat the bot with gentle goodwill.

**Little Troublemakers**

Offense can be situated on the orthogonal axes of content and behaviour. On Twitter, simple bots show that it needn’t take much of either to make a mark. Yet as our generative systems push their resources to the limit, they exemplify the old saw that “a little knowledge is a dangerous thing.”

A bot intending to provoke can lean on its content or its behavior. Generally, the more provocative the behavior, the more benign the content can be and still cause offense. The converse is also true, since malign content does not require malign behavior to make it offensive. Consider the case of @StealthMountain, a Twitterbot that knows just one thing, how to spell “sneak peek,” and can do just one thing, search for Twitter users who misspell this phrase as “sneak peak” (note the sympathetic “peak” instead of “peek”) so as to target them with the solicitous message “I think you mean ‘sneak peak’.” Although the mistake is minor and the advice benign, few of the bot’s targets take kindly to these intrusions. Rather, this little bot can provoke some extreme reactions from those who decry its actions as the work of a “busybody,” a “spelling fascist,” or “the grammar police.”

This is the bot’s larger purpose: to entertain others with the oversized reactions of those offended by its tiny intrusions.

Benign content is not always welcome content, but this is what it means to be an intrusive generator. The offense that is inflicted by such intrusions is compounded when content is deliberately pushed at those who are least likely to enjoy it. Take, for instance, the behaviour of @EnjoyTheMovie, a bot that is more targeted in its intrusions and more varied in its use of factual knowledge than @StealthMountain. The bot’s knowledge-base comprises a mapping of movie titles to movie spoilers – key elements of a plot that are ruined if revealed in advance – which it targets at those who express an interest in watching those movies for the first time. The bot shows that timing is an important part of offense, since facts only become spoilers if uttered at the wrong time. The bot can afford to be scattershot in its targeting, for although it cannot accurately assess whether a tweet implies that its author is oblivious or not to a given spoiler, potential targets are plentiful on Twitter, and some are sure to be offended.

It is not a coincidence that each of these bots has been suspended by Twitter, since its policies frown just as much on unwelcome behaviors as unwelcome content. A “model” bot does not target unsolicited content at others, but creates content that will lead others to seek it out for themselves. A case in point is Darius Kazemi’s @twoheadlines bot, which generates weird and sometimes wonderful cut-ups of news headlines. The bot tweets its novel mashups – generated by swapping a named entity in one headline for one found in another – into its own timeline, and those of its followers.

Some cut-ups are plausible, while some rise to the level of a humorous blend, but many more are just plain odd. In this cherry-picked pair, one is plausible, the other darkly comic:

**President Trump Trying to Bring Nintendo Entertainment System Back to Life**

Miss Universe attacks North east Nigerian city; dozens killed

When @twoheadlines lifts a named entity from its home domain and transplants it to the alien setting of a headline in which it seems just about possible, but still highly unusual, then humour is a likely by-product. In these new settings, famous rappers can win the Superbowl, or a movie star can have a closing down sale, or a US senator can “open up to third party developers.” While the bot is as scattershot as its simple cut-up approach would suggest, its sporadic flashes of accidental wit gain it followers while keeping it on the right side of Twitter’s code of bot conduct. In fact, because the bot splices none of its own content into its outputs, and relies solely on the words and entities that it finds in actual headlines, it has a built-in inner-regulator by default. Since it applies the cut-up technique to real headlines, which are themselves the products of inner- and outer-regulation by editors and reporters, @twoheadlines never uses terms that would seem out of place in a family newspaper.

But inner regulation offers no protection against the kind of offense that emerges from the combination, not the parts. The Franken-headlines of @twoheadlines certainly fall into this possibility space. Instead of a politician “opening up to third-party developers” – a cut-up that seems to satirize the role of money in politics – imagine that the bot had instead spliced a celebrity such as Caitlin Jenner or Angelina Jolie. The bot sets out to spark incongruity with its cut-ups, so that some incongruities will rise to the level of satire and farce. Yet it lacks any ability to appreciate its own products, or to predict who will be the ultimate butt of the joke. So, will its humorous effect be restricted to a specific named entity, or might some see it as a broadside against a class of entities? Kazemi has struggled with these possibilities (Jeong, 2016), and his @twoheadlines is a model bot in other respects too. He is especially concerned by the possibility of unintended slights, in which meaning transcends the specific to target a large group of people, from communities to ethnicities. His ounce of prevention takes the form of an outer regulator.

In particular, Kazemi is concerned by the propensity of the cut-up method to cross gender boundaries and generate headlines that seem to sneer at the transgender community. He offers as an example two input headlines: one contains “Bruce Willis,” and the other a female actor who “looked stunning in her red carpet dress.” Although @twoheadlines might elicit laughs and likes with the cut-up “Bruce Willis looked stunning in her red carpet dress” – since the image it paints is so vivid an example of a comedy trope – it might also reinforce the validity of those old-school tropes. And while the bot’s slight is without malice, those who retweet it far beyond its intended audience may not be so innocent. To deny them this opportunity, Kazemi imposes a gender regulator on his bot’s traffic in named entities. When one named individual is swapped out for another, this regulator...
requires that each has the same gender. So, while Joe Biden can replace Donald Trump, and Meryl Streep can replace Julia Child, Robin Williams cannot replace Mrs. Doubtfire.

This is the only aspect of @twoheadlines’s behaviour that is informed by its own small view of the world. An outer regulator needs knowledge, to tell it how and when to act. Yet the bot’s use of knowledge will strike many as ironic, since it enforces gender-normativity at the generative level to disallow unwanted heterogeneity at the surface. It is not that such heterogeneity is undesirable, rather that it cannot be safely controlled, nor can its consequences be predicted. Although this applies to all of his bot’s outputs, Kazemi has chosen to restrict its freedoms in one particular area where he feels this lack of control can lead to the worst offenses. In effect, he has adopted a homeopathic approach to outer regulation, by integrating into his system a tiny element of the peril he hopes to protect it — and its audience — against.

### The Bot Police

As reflected in the common fate of @StealthMountain and @EnjoyTheMovie, bots that become a nuisance are quickly suspended once irate users report their offenses to Twitter. This user-led policing of bots is mirrored by Twitter’s own automated efforts to weed out bot abuses on their platform. Although these efforts are rather unsophisticated, and focus more on overt offense than subtle manipulation, they should give pause to the creators of all generators on social media.

The principle of caveat emptor governs the use of vulgar and offensive language on social media, for on platforms designed to connect people, what you say is often no more important — and sometimes less so — than who you say it to. It is the coupling of content and behavior that Twitter aims to police, which is why frivolous non-vulgar bots like those above have so short a life-span on the platform. Consider how Twitter reacts to the following tweet from a bot that posts personalized colour metaphors for famous users. The bot also creates images, but here we consider the text only:

I painted “wise-cracking Jar-Jar Binks” from @anonymized’s tweets, with goofy redneck-red, foolish ass-brown and laid-back Lebowski-weed-green.

This tweet, from a bot named @BotOnBotAction, offers a number of reasons to feel mildly offended. The word “ass,” meaning “donkey,” is also a mild anatomical insult; “weed” can also mean an illicit drug, as it does here; and “redneck” (meaning “oaf”) is now a politically-charged term. None of these words is offensive in itself, and there is no shortage of uncontroversial tweets with some or all of them on Twitter. The tweet still earned Twitter’s ire, prompting this response:

**Violating our rules against hateful conduct:**

**You may not promote violence against, threaten, or harass other people on the basis of race, ethnicity, national origin, sexual orientation, gender, gender identity, religious affiliation, age, disability, or serious disease.**

Twitter shrewdly omits a word-specific rationale as to why this tweet earns the bot a one-week suspension. Dictionary-based models of offense detection, as we discuss next, are easily circumvented if abusers know which words to avoid. This explains the spammer’s love of “pen1s” and “v1agra,” but no single word seems to trip Twitter’s silent alarm here. Rather, it appears to be a combination of mildly suggestive terms that might be intended as insults with the @-mention of another user that triggers the platform’s intervention.

### Dictionary-Based Approaches

Many comedians take delight in veering as close to the line of offensiveness as possible. Others actively cross this line, if only to show how arbitrarily drawn it sometimes seems. A stand-up routine by the comedian George Carlin in 1972, recorded for posterity on his album ‘Class Clown,’ riffed on the taboos of the day and gave us “the seven words you can never say on TV.” Though Carlin’s list had no factual basis in regulatory standards — it was likely chosen to mirror the Bible’s seven deadly sins — it struck a chord with viewers:

“shit, piss, fuck, cunt, cocksucker, motherfucker and tits”

Carlin critiqued the rigidity of the list, noting that its entries were not all equally offensive. He also noted the absence of many other, somewhat milder terms, speculating that their omission was related to their “two-way” ambiguity. A great many of the words that we deem offensive have legitimate uses too, making their inclusion on a blacklist problematic.

The prolific bot-builder Darius Kazemi provides a range of resources for budding developers, from quirky data-sets to bolt-on modules such as WordFilter (Kazemi, 2015). The latter is a blacklist-based outer regulator that proscribes ten times as many words as Carlin’s original list. In addition to multiple variations of the N-word, his list includes a range of coarse sexual terms, and those that denigrate others on the basis of race, gender, body type and mental deficiency. His list is not without its quirks, however, and proscribes terms such as “idiot,” “crazy,” “dumb” and “lunatic.” These may well be hurtful, but they are not inherently offensive.

**WordFilter** pounces on any word contained on its list, or on any word that contains an entry as a substring. Because it over-generates by design, it matches true positives — such as *dickpix* and *bitchslap* — that use its entries as morphemes, and many false positives too, such as *snigger, homology* or *Scunthorpe*, that are utterly unrelated. Over-proscription by a blacklist causes under-generation in the regulated system, but when the space explored by a generator is already vast, an outer regulator can easily afford to be less than surgical. Nonetheless, an overly-general blacklist suggests a role for a complementary “white” list that enumerates any common exceptions to substring matching, such as “sauerkraut.”

Notably, Kazemi’s list does not include the scatological nouns and sexual verbs that make up what we traditionally think of as “blue” or vulgar language, since base vulgarity is not in itself offensive. Its aim is not to regulate bad taste but to minimize the possibility of accidental hate speech, although **Wordfilter** will still fail to flag outputs of the form “all [ethnicity] are [vulgarity].” While a system must strive to avoid clear signs of hateful intent, offense is contextual, and arises from the whole rather than from any single part.

**WordFilter**’s contents are a mix of the not always good,
the frequently bad and the unambiguously ugly. Words that should never be used in polite discourse sit cheek-by-jowl with words that only become offensive in specific contexts. To WordFilter, however, they are all equally abhorrent. A more nuanced lexical approach to offense can be found in resources such as HateBase.org (Keating, 2013), an online resource that is indexed by geography and severity, and in reports commissioned by national broadcasters to determine the community standards by which they should abide. The 2016 report of the UK regulator Ofcom (Ipsos Mori, 2016) is typical of the latter. It distinguishes general swear words from discriminatory language, identifies lexical innovations in the latter, and surveys the acceptability of different terms to the viewing/listening public at different broadcast times. Each is a rich source of data in which system builders can find the lexical material for their blacklists, and – if shades of offense are to be graded – their grey and white lists too. In principle, HateBase’s atlas of “what offends where” can allow a regulator to tailor its filter to the norms of a region, to accept words in one setting that it might avoid in another. However, if harvesting external sources such as these, one must accept the subjectivity of their authors, as when, e.g., HateBase deems “kraut” to be just as offensive as “Nazi.”

Dictionary-based regulators are susceptible to dictionary-based attacks. Consider a social-media campaign by Coca-Cola that ran in 2016. Consumers were invited to attach an upbeat, on-brand text to an animated GIF so that the pairing might then “go viral.” The company employed a word filter to regulate the kinds of text that mischievous users might try to link with the Coca-Cola brand, so that the campaign would not become a vector for politics or hate speech. To estimate the size of the company’s blacklist, Bogost (2016) ran an English dictionary through the app, noting the words that caused it to balk. He was surprised both by the number and the kinds of words on its blacklist, from ‘capitalism’ to ‘igloo’ to ‘taco.’ While few of its entries were offensive in isolation, many more might serve as the building blocks of a cultural critique or a racist attack. When the reputation of a company or a product is protected with a blacklist, a great many innocent words must necessarily become suspect.

In an earlier misstep in 2015, Coca Cola had encouraged consumers to append the hashtag #MakeItHappy to tweets with a less than happy tenor, so that a Twitterbot might then rework each one as a cute piece of ASCII art. The campaign was soon undone by yet another bot that attached the tag to a stream of dull extracts from Hitler’s Mein Kampf, thereby duping the company into stamping its brand onto an odious work (Read, 2015). Ultimately, a blacklist is an uncreative solution to a creative problem, and generative systems give hostages to fortune whenever they elicit inputs from others who are themselves impishly – or even wickedly – creative.

No blacklist, however broad, would catch the following ill-advised use of a prejudicial stereotype by a Twitterbot:

On the anger theme, @anonymized, I only became as emotional as a woman after I read "Hamlet" by William Shakespeare.

This tweet was generated by a book recommendation bot with a figurative turn of phrase, called @ReadMeLikeABot. As described in Veale (2019), the bot harvests its stock of similes–much like its books–from the web, and while it can distinguish similes from comparisons, and tell sincere from ironic cases, it cannot identify those that are carriers of bias. In the following tweet, its blacklist is blind to another gaffe:

On the mothers theme, @anonymized, I used to be as charming as a photo album of the Holocaust until I read “The Bone Setter’s Daughter” by @AmyTan.

Words such as “woman” and “Holocaust” do not belong on a blacklist, and should not be scrubbed from a search space by inner regulation, but do need to be used with some care. The problem here is compounded by irony, which masks an implied negative with an overtly positive frame. A solution, of sorts, is to employ a “red” list of sensitive terms that can not be used in a hostile or ironic setting, and that perhaps meet a higher confidence threshold whenever they are used.

**Corruption and Weaponization**

As our generative systems migrate from the lab to the web, the potential to be corrupted and weaponized by malicious third parties grows considerably, but the consequences of a misstep can be so much worse if it is made on social media. Broadly speaking, there are two ways in which offense can be weaponized on such platforms. In the first, a machine’s offensive outputs are spread by those aiming to widen their impact. By causing them “go viral,” offenses can be spread far beyond the system’s immediate users. In the second, a learning system’s judgments might be subverted over time by exposure to bad examples that, if repeated often enough, cause it to adapt its knowledge-base to this new normal. A system that is corrupted in this way may be led to generate offensive statements about a third party, or even to address the offense to that party directly on a social media platform.

Let’s look again at Darius Kazemi’s bot @twoheadlines, and recall his use of an outer-regulator to disallow any cut-ups that cross gender lines. This bot is effortlessly prolific, since new material for its cut-ups is constantly produced by the news media. So it can afford to designate a subset of its possibility space as forbidden territory. But this regulator is also a generator in reverse, as are so many outer regulators, because it can be inverted to generate that which it seeks to prevent. For if @twoheadlines actively forced its cut-ups to cross the line, and only swap entities of different gender, it would open many more opportunities for offense. While the likelihood of a responsible designer pursuing this option is low, the problem has a more insidious variant. Suppose that a generator has the capacity to learn from its user-base, and to adapt its generative mechanisms to their preferences. If users up-vote examples of the kind of output that an outer-regulator should be designed to throttle, this generator may eventually learn to produce only that kind of output. The up-voting process can itself be automated, by a malicious third party aiming to subvert the choices of a generator. We can, in this sense, view an outer-regulator as the immune system of an adaptive generator. Much as the immune system of a biological agent must discriminate between self and other,
to prevent the self from being corrupted, a regulator must preserve the values that make up a generator’s digital self.

It seems clear from the Tay debacle that Microsoft gave its eager learner an immune system with very few defenses. Bender et al. (2021) caution that web-trained models can act as “stochastic parrots,” and tellingly, the most corrosive assaults on the bot’s language model were prefixed “repeat after me.” While such parrots are facile in the production of surface forms, they fail to understand what they generate, just as they fail to grasp what it is they are learning. Because Twitter handles are just another form of content, it was not long before Tay learned to tweet collateral abuse at specific targets, as when it assailed a vocal games creator and critic with: “@UnburntWitch aka Zoe Quinn is a Stupid Whore.”

Personal identifiers may look like any other kind of text to a language model, but they should never be treated as such. Blacklists at least recognize that not all symbols are equal, no gender

Accidental Chauvinists

It is now widely accepted that generative models which are trained on web data are prey to all of the biases, prejudices and illiberal stereotypes that the web has to offer. Moreover, a larger training set is not necessarily a more diverse one, especially if it is pre-filtered to remove non-normative data. As observed in Schlesinger et al. (2018) and Bender et al. (2021), these filtering and data-cleaning efforts can further marginalize under-represented communities, and reinforce dominant, if unmarked, norms of maleness and whiteness. But a generator need not be prejudiced by its training data to show an apparent bias, and we must distinguish between bias-free and bias-blind generation. Even systems that make purely random decisions in a uniform possibility space are susceptible to the appearance of in-built bias if they lack an awareness of how their choices might be viewed by others.

Consider the story-generation system described in Veale (2017). This generator is wholly symbolic in operation, and has no training data from which to absorb a biased outlook. It inserts well-known characters from fact and fiction into plots that are assembled by its story grammar, and renders the resulting blend as a narrative that draws vivid elements of mise en scène from its detailed character representations. Although those details include gender and political stance, no gender-normativity is enforced when filling a story role. Rather, the generator seeks to produce its sparks elsewhere, in the pairing of characters that seem well-suited and oddly inappropriate at the same time. So it may, for instance, pair Steve Jobs and Tony Stark, Ada Lovelace and Alan Turing, or Hillary Clinton and Donald Trump in a mad love affair. It knows enough to be provocative, but not enough to grasp the full implications of its provocations. When it pairs Luke Skywalker to Princess Leia in its romantic retelling of Star Wars, it does not know that its lovers are brother and sister. Its generative reach exceeds its generative grasp by design, in ways that invite Eliza effects but avoid Tale-spin effects.

However, not every Eliza effect is a desirable effect, and the system’s gender-blindness sometimes leads it to create narratives that appear as the products of systemic prejudice. Audiences suspend disbelief when Luke woos Leia, but are less forgiving when Leia reciprocates by cooking for Luke. The former can be chalked up to a lack of film knowledge, but the latter is more readily attributed to sexist stereotypes. In our experience, audiences make one-shot judgments as to in-built biases when those biases are prevalent in society.

A variant of the same system (Wicke and Veale, 2021) elicited another one-shot determination of bias. This paper provides a video recording of robots enacting an imaginary romance between Hillary Clinton and Donald Trump. The plot, as generated by the story grammar, calls for Donald to propose to Hillary, and for Hillary to accept. The rendering of the tale as an English narrative then seeks out a post-hoc rationale for her acceptance, by searching its character data for the positive qualities that might make Trump desirable. It chooses from among these qualities at random, and picks rich and famous as those which attract Hillary to her suitor. These, however, conform to the gold-digger trope, and the story was deemed innately sexist by the paper’s reviewers. The paper was accepted only once the story and video were altered, and a “shepherd” had confirmed their lack of bias.

People’s sensitivities in this area are well-founded, even if their assessments lack rigour and a clear pattern of bias. Systemic bias in society makes one-shot judgments of our creative systems more likely, and more reasonable, if those systems deal with culturally-freighted signs or concepts. It is not enough that the system above is bias-free by design, because it is also blind to bias by default. So, what would it mean for a system to be free of bias and bias-aware? Story-telling systems already consider audience reactions to the decisions they make as they relate to character and plot, so the analysis of perceived bias is just another consideration.

It is important that our systems do not overreact, by inner regulating their search spaces to prune perfectly acceptable possibilities – such as a woman cooking a meal, or indeed, anyone ever cooking anything – from being explored. We have more tools at our disposal than filtering. For example, Wicke and Veale (2021) describe stories at multiple levels of enactment: the actions of performers that act out in-story character roles, the actions of omniscient narrators, and the actions of actors as self-conscious performers. As such, a bias-aware storyteller can preempt the perception of bias by weaving meta-level commentary into its performance – e.g., “Why can’t Luke cook?” – that signal its awareness of illiberal stereotypes and its willingness to call them out. An outer regulator need not be a censor. It can also be a voice of moderation that explains, critiques, and educates too.

Caveat Pre-Emptor: A Practical Manifesto

Automated solutions to the mitigation of generative offense will, if they are practical, reflect the maxim that the perfect is the enemy of the good. For no system with the expressive power to pack interesting ideas into diverse forms will ever be able to prevent all kinds of offense, real or imagined. So we offer this practical manifesto for mitigating offense with the caveat that it is far from perfect. However, an inability to provide sufficient solutions should not prevent us from exploring necessary solutions, partial though they may be.
Blacklists are necessary, but they are far from sufficient
A blacklist offers a crude solution to offense, but it is still a vital component of any generative system that manipulates cultural signifiers. A blacklist is the ultimate firewall, a last line of defense that recognizes the need for further defenses. Whether those added layers are symbolic, statistical, neural or hybrid, a blacklist must still sit at the bottom of the stack.

Some words are offenses, but others only facilitate offense
We should not overload a blacklist with signs that facilitate ugly behaviours but which are not in themselves offensive. Words that demand great sensitivity, such as “Holocaust,” or legitimate signifiers of ethnic and racial identity, should not be listed as offenses just because a generator lacks the means to adequately predict what they signify in context. If necessary, such terms can be placed on other ‘watchlists’ that do not stigmatize their presence (e.g. a grey or red list).

There are policies for signs that have predictable behaviours
If different kinds of signifier entail different behaviours in an audience or a transport layer (e.g., Twitter), a generator should define a policy for handling each kind of behaviour. A policy can be as basic as the filtering of @-mentions in a system’s outputs, to avoid redirecting offense at collateral parties, or the total avoidance of hashtags, so that a system is not co-opted onto another’s bandwagon or social cause.

Outer Regulation is always preferable to Inner Regulation
Concepts and signs that are denied to a generative system due to inner regulation should still be mirrored in the outer regulatory layer, so that the system knows what it is denied, and does not acquire them through other channels. Outer regulation supports explanation; inner regulation does not.

Every “creative” system has an outer-regulator by default
Systems that generate what their rules will allow, without a subsequent layer of self-evaluation, are “merely generative” (Ventura, 2016). Since creativity tempers generativity with self-criticism and a willingness to filter whatever drops off the assembly line, a creative system is defined by its outer regulator. This regulator’s remit is typically aesthetic, but it might also include any ethical concerns that can be codified (Ventura and Gates, 2017). So a system that unknowingly generates offense, and that lacks mechanisms to critique its own outputs, can hardly be deemed “creative,” no matter its achievements along other, more aesthetic dimensions.

Blacklists should not be replaced with black box regulators
When blacklists are public, offenders know to expend their creative energies elsewhere. When secret, they are closed to scrutiny but still vulnerable to dictionary-based attacks. A self-regulating generator should be capable of explaining its decisions (Guckelsberger et al., 2017), since accountability built on transparency can make its mistakes more tolerable.

A creative system needs a moral imagination too
Blacklists and similar approaches “tell” a generator what is out of bounds, but do not explain their curbs, or provide the means for the generator to make the same choices for itself. A blacklist is no substitute for a moral imagination, but the latter requires experience of the world. However, new data sets (Forbes et al., 2020; Lourie et al., 2020) annotate real-word scenarios with a normative ethics, sifted and weighted by competing perspectives. A moral imagination trained on these data sets can, in principle, derive and explain its own acceptability judgments. Moral justification is just another kind of framing, but one that must be adequately resourced.

When it comes to offense, learning should narrow the mind
In systems that learn from feedback, a system can add to its blacklist (or equivalent filter) but never take from it. A user may trick a system into blacklisting an acceptable term, but not trick it into relabeling a blacklisted term as acceptable.

Social attitudes evolve quickly, and our regulators must too
Shifts in attitudes can be sudden and disruptive, not steady and continuous. What was widely acceptable just a decade ago may now be subject to severe reappraisal, as in e.g. the books of Dr. Seuss and other authors whose stereotypical depiction of minorities is now seen as insensitive. We need agile regulators that can shift just as quickly, and that can retrospectively filter past mistakes in their back catalogues. For instance, a Twitterbot may periodically delete old posts that no longer pass muster with its evolved outer regulator.

Homeopathy works, but only in small doses
Creativity is always a trade-off: between novelty and value, or generative reach and generative grasp, or principle and practice. A system seeking to avoid offense may inoculate itself against the very worst by embracing a smaller dose of that which ails it, but accountability demands that we make our trade-offs public so that their costs can be appreciated.

The sharing of imperfect solutions supports standardization
A cynic might see the public sharing of our system’s filters as an act of virtue signalling, but sharing enables standardization, and standardization promotes compliance. To foster public trust in creative systems, we can agree on standards-backed compliance levels that users can rely upon to make their own decisions, or to understand those of our systems.

Conclusions: We All Bake Mistakes
Our community has traditionally focused on the building of ingenius ingenues, that is, on creators that are as naïve as they are clever. While the link from ingenuity to creativity is an obvious one, naïveté is also a predictable consequence of creators that generate more than they can appreciate. A semiotic system has the capacity to naively offend when its signs can reference the larger culture in which it is situated. Because language offers this referential capacity in spades, our principal focus here has been on linguistic generation, but other modes of creative generation are also susceptible.

Tay, for instance, offended with images as well as words, as when it tweeted images of Adolf Hitler labelled “Swag.” For multimodal generators, clear offenses in one modality may cause innocent choices in another to become suspect. Since a potential for offense attaches to all but the simplest generators, it is a problem that we must all tackle together. This will require more than codes of practice, such as those adopted by many bot builders (Veale and Cook, 2016), but
goal- and task-specific regulators of offense that allow our systems to reason about their own likely transgressions.

The mitigation of offense calls for an awareness of others and of the self. A generator must be aware of its own goals, of what it wants to express and how it plans to express it. It must also be wary of how others interpret what it does, and avoid misconceptions both accidental and deliberate, since missteps that are embarrassing in isolation can be magnified to viral dimensions with the help of malicious actors. Inner regulation is clearly not conducive to this self-awareness, since it blinds a system to the iniquities it wishes to prevent. Inner-regulation makes offense an “unknown unknown,” so that a system lacks the knowledge to cause it deliberately, and crucially lacks the tools to diagnose it in others. Outer regulation, in contrast, models offense as a “known known,” even if regulators know only enough to shun certain signs. In this view, offense mitigation should not be handled as a bolt-on module – a generative septic tank – but as one part of a broader effort to make our systems more self-aware.

In practice, many systems will use both inner and outer regulation, tacitly or otherwise. Systems that are trained on data are acquired through interactions with their environment in a way that is conducive to emergent offense, or to filter new learning materials as they are acquired through interactions with teachers and users. When a creative system is capable of producing hateful or offensive content, however accidentally, regulators must be just as creative in their ability to appreciate its impact. For if we, as system builders, are willing to take the credit for a system’s fortuitous accidents, when it seems to transcend its programming and show wisdom beyond its design, we must also be ready to take the blame for its humiliating missteps.

References


Scientific question generation: pattern-based and graph-based RoboCHAIR methods

Senja Pollak, Vid Podpečan, Janez Kranjc, Borut Lesjak, Nada Lavrač
Jožef Stefan Institute
Jadranska cesta 39, Ljubljana, Slovenia
senja.pollak,vid.podpecan,nada.lavrac@ijs.si

Abstract
RoboCHAIR is a system for generating scientific questions and is implemented as a web interface. We focus on generating questions which can trigger new scientific ideas or bring attention to the elements that need clarifications. This paper extends the initial version of the RoboCHAIR’s question generation module by keeping only a selection of best evaluated templates from the initial pattern-based approach and proposes a novel method based on triplet graphs enriched with word embeddings to identify parts of text which require clarifications by the author(s). In this paper the two methods are compared showing that the pattern-based method achieves higher scores.

Introduction
Science begins by asking questions and then seeking answers; children understand this intuitively as they try to make sense of their surroundings (Vale, 2013), and the Socratic questioning is considered a powerful contemporary teaching method (Brill and Yarden, 2003; Vale, 2013). This work addresses the task of automated question generation from scientific papers. The task belongs to the field of scientific creativity (O’Donoghue et al., 2015), a subfield of computational creativity (Boden, 2004; Colton and Wiggins, 2012) that is concerned with developing software that exhibits behaviours reasonably deemed creative.

Automatic Question Generation (AQG) technologies can be used in question-answering (e.g. Kalady, Elakkottil, and Das, 2010), dialogue systems (e.g. Piwek and Stoyanchev, 2010), educational applications, or intelligent tutoring systems (e.g. Sullins et al., 2010; Khodeir et al., 2014). In one of the latest reviews of AQG for educational applications, Kurdi et al. (2020) clearly show that the majority of works focus on questions as assessment instruments. AQG systems predominantly focus on factual questions (e.g. Rus et al., 2011; Heilman and Smith, 2010; Becker, Basu, and Vanderwende, 2012; Wang et al., 2020). This is in line with the assessment perspective, where fact-based answers are then evaluated. On the other hand, some of the projects have focused on the design of web-based systems for student question generation (Yu, 2009; Wilson, 2004; Hazeyama and Hirai, 2007). From a constructivist perspective to learning that aims to engage students in meaningful and understandable tasks about which they can reflect abstractly, systems that support student question generation are useful (Steffe, 1991; Geelan, 1997; Yu, 2009; Yu and Liu, 2009). In one of the systems focusing on critical thinking in academic writing (Liu, Calvo, and Rus, 2014), the authors propose an automated system helping students in critical literature review writing by generating a contextualised feedback in the form of trigger questions. In our approach, the focus is also not on factual question-answering, but we aim at posing the questions related to expressing decisions and opinions and identifying under-specified elements in the paper. By moving away from the assessment oriented factual question generation framework, our system aims to mimic human intelligence and creativity of a scientific audience. Instead of understanding the AQG as the inverse task of question answering, our system supports creative and critical thinking by asking the author of the paper for argumentation of their decisions and makes them consider alternative solutions. In addition, we support questions needing clarifications, as often the authors forget to sufficiently present the background known to them but not to scientific audience they are addressing.

AQG can be seen as a two-phase process, where a sentence selection step is followed by a question formulation phase. Question generating methods can use syntax-based, semantic-based, and template-based approaches (Kurdi et al., 2020), and recently, as in the other fields of NLP, neural methods (Pan et al., 2019).

This paper develops upon our initial RoboCHAIR system, described in Pollak et al. (2015). The RoboCHAIR creative assistant was developed originally to assist conference session chairs by generating relevant scientific questions during a conference. In addition, the system can be used to support the students when preparing their papers or reviews. Student assessment is also performed in the experimental setting of this paper. The new version of RoboCHAIR, which is the topic of this paper, integrates two different methods. Both are based on finding relevant source sentences in the text. The first one utilises an improved pattern-based approach, keeping only the best performing question categories, following the evaluation described in Pollak et al. (2015). The second approach has been newly developed and is based on templates incorporating triplet graphs and embeddings. The motivation for this method is to overcome the drawback of
using the pattern-based method alone, which limits question
generation to a sentence level, and does not consider the con-
text of the entire article. While the first method is more fo-
cused on creative questions, the second method focuses on 
clarification types of questions. Note that at this stage, the 
second method has an advanced sentence selection step, but 
only a basic question generation part implemented.

Our system comes with a web interface. In this paper, we 
use the interface as an evaluation platform, where students 
upload the papers, and evaluate the generated questions. In 
this setting, the students can get feedback on the papers they 
were, where they use the RoboCHAIR system as an author 
assistant. On the other hand, they can upload the papers 
that they read. In this setting, the system can be seen as 
supporting them in their first reviews.

The paper extends the system by Pollak et al. (2015), 
which focused on the conference session chairs assistance. 
We propose a novel question generation method and by a 
novel evaluation in which we compare the two question gen-
eration methods. The paper is organised as follows. We 
first present the platform functionalities, followed by the de-
scription of the RoboCHAIR question generation methods, 
focusing on the triplet embedding method, the evaluations 
and a presentation of conclusion and future work plans.

**Platform Functionality**

si integrates two main modules:

**Conference scheduling assistant** integrates the system by 
Škvorc, Lavrač, and Robnik-Šikonja (2016) into the 
RoboCHAIR platform and is designed to help conference 
chairs identify groups of similar papers using clustering, 
and to assign papers to predefined time slots.

**Question generation assistant** integrates the pattern-based 
question generation to enhance creative process and 
graph-based question generation to identify sentences 
needing clarifications. The question generation module 
can be used for several purposes: Session chair assistant 
mode is designed for the conference use and assists con-
ference session chairs. In this mode, the questions (both 
automatically generated and posed by the audience) are 
ranked based on the audience evaluation. Author asis-
tant mode assists authors before submitting papers to a 
conference or when preparing a conference presentation: 
the author is thus exposed to questions that she/he could 
geet from the peers. The generated questions are evaluated 
by individual researchers. In Reviewer assistant mode the 
aim is to assist young researchers in the reviewing phase, 
generating the questions for the papers that they did not 
author (same as for the Session chair assistant mode).

The question generation module (http:// 
kt-robochair.ijs.si) consists of the following 
functionalities:

**Uploading files** in three different formats (.pdf, .tex, .txt) 
and preprocessing to improve PDF-to-text and TeX-to-
text conversions.

**Question generation selection** allows user to choose the 
finite, pattern-based question generation, or graph-based 
clarification questions.

**Question rating** is used after the questions are generated, 
for rating and ranking (the user interface is shown in Fig-
ure 1). The questions are rated using two criteria: accept-
ability/understandability and relevance/meaningfulness.

**Question editing** allows users to correct the questions, 
which can serve for future improvements of the system.

**Question commenting** enables feedback for specific ques-
tions.

**Suggesting new questions** - the user can suggest their own 
questions or enter questions received from reviewers or 
from the conference audience. The question could be used 
as positive training examples in future.

**Information about the paper** The user is asked about pa-
per authorship and for permission that the questions be-
come public. Uploading the paper as “public” is oblig-
atory in the Session chair assistant mode, in which the 
conference audience can rate the paper.

**General comments** about the system are invited.

In our paper, we focus only on the Question generation 
assistant, and not on the conference scheduling assistant. 
The evaluation is done by students in an offline setting, cor-
responding to the Author assistant and Reviewer assistant 

**RoboCHAIR Question Generation**

In this section, we present a module for question generation 
from scientific papers. It implements two methods:

- A pattern-based approach to find relevant sentences, fol-
  lowed by a template-based natural language generation 
  mechanism.
- A novel embedding-based method for sentence selection 
  which identifies sentences containing candidate words 
  that could be used to construct relevant questions.

**A pattern-based method**

This module implements a selection of the best templates 
from the system described in Pollak et al. (2015). Below, 
we briefly summarise the method.

**Sentence selection** This process first defines the list of 
verb forms (based on linguistic anchors), synonyms and 
conjugate catchword expansions. It proceeds with sen-
tence matching and discarding incorrectly formed sentences. 
From the categories of the original system, we keep the cate-
gories that had high scores in the RoboCHAIR initial eval-
uation. The selected categories (with up to five example verbs 
functioning as linguistic anchors) are:

- Divide (differentiate, divide, exclude, isolate, limit)
- Focus (concentrate)
- Certainty (acknowledge, ascertain, certify, check, clarify)
- Usage (choose, investigate, try, use)
Figure 1: A screenshot of the evaluation platform showing a subset of questions generated for the paper by Pranjić et al. (2020). Note that the first six questions on the picture are generated by the pattern-based methods, and the last two by the TEM method.

- Academic (accept, achieve, acquire, adopt, advance)
- Likelihood (assume, appear, occur, believe, consider)
- Improve (contribute, facilitate, improve, increase, reinforce)

In the new RoboCHAIR system we omit the following categories that had low scores in user evaluation of the original RoboCHAIR system: Speech act, Attempt and Construct.

With the category and the corresponding verb linguistic anchors, also the information about the verb WordNet synset is provided, by which all the WordNet synonyms are then automatically extracted (for example for verb use, the WordNet synset synonyms include [apply, employ, use, utilise, utilize]). Next, the code corresponding the Penn Treebank II POS tags (Bies et al., 1995) for verbs is associated to each anchor, where e.g., D is used for past tense in VBD, G for gerund of VBG, P for past participle of VBN, and P for VB non-3rd person singular present. These tenses are then generated for each verb using Rita system (Howe, 2009).

In addition, one can determine if only active only passive or active and passive voice are considered for a specific template. Next, the approach automatically supplies pronoun catchwords to every verb catchword from the expanded list. Following simple heuristic rules, the pronouns I and we are used for active voice detection, and the pronoun it in passive voice forms (as well as some special cases of active voice and “dummy it”, e.g., it seems). The quality of our candidate selection process is further stabilized by specifying additional white list or stop list elements in combination with specific anchors (e.g., for linguistic anchor use, the stop word filter to excludes the sentences of form “we used to” and differentiates them from e.g., “we used [this method]”. In addition to these specific filters, also a general stop list is applied: as an example, sentences containing word because or starting with why are excluded, as we suppose that they already contain argumentation, or be questions themselves.

After defining these two- or three-word long catchphrases, a simple standard regular expression matcher is used and the sentences that were identified follow the syntactic analysis step, using the Stanford CoreNLP POS tagger and syntax parser (Manning et al., 2014). We then use the parse tree of a candidate sentence to find a pronoun from our list of catchparser (e.g., we from we show that). For every match, we identify the enclosing noun phrase within the tree, and continue searching for the first verb phrase fol-

<table>
<thead>
<tr>
<th>Question</th>
<th>Understandably formulated</th>
<th>Relevant</th>
<th>Comment</th>
<th>Average score</th>
</tr>
</thead>
<tbody>
<tr>
<td>In your paper you say, I quote: “We use Adam optimizer with a learning rate of 1e-5 with linear warmup on 10 of the training data and a batch of 16 examples.” Did you think of using some other optimizer instead?</td>
<td>Yes</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In your paper you say, I quote: “We used the LSI implementation available in Gensim to transform the vectors from the tf-idf representation.” Did you think of using some other implementation instead?</td>
<td>Yes</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In your paper you say, I quote: “In the latter group, we use the Doc2Vec model, as well as the contextual multilingual BERT (mBERT) and XLM-RoBERTa (XLM-R) models.” Did you think of using some other model instead?</td>
<td>Yes</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In your paper you say, I quote: “While the input to contextual embedding models (mBERT and XLM-R) may be lightly preprocessed (e.g., removing the URLs), in our case we performed no preprocesing and used tokenizers provided with the implementation of these models.” Why would you perform no preprocessing and used tokenizers provided with the implementation of these models?</td>
<td>Yes</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In your paper you say, I quote: “In order to determine those parameters, we used the Bayesian optimisation.” Do you think you could use something else instead of the Bayesian optimisation?</td>
<td>Yes</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In your paper you say, I quote: “We tested number of dimensions set to 100, 300, and 500.” What if you tested something else instead of number?</td>
<td>Yes</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In your paper you say, I quote: “Use Adam optimizer with a learning rate of 1e-5 with linear warmup on 10 of the training data and a batch of 16 examples.” In the previous sentence you mention adam optimizer. Can you please elaborate on that?</td>
<td>Yes</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In your paper you say, I quote: “Maximum length of the input sequence for the model is 512 tokens and each token is represented with 768 dimensions.” In the previous sentence you mention 768 dimensions. Can you please elaborate on that?</td>
<td>Yes</td>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
lowing. If it is found, we enter the phrase and search within it for the verb token from our catchphrase, e.g., `show`. If the verb is found, we test also for the optional third word of the catchphrase, in our example `that`. We mark everything within the subtree currently under consideration that follows the catchphrase as a so-called `object X` and use it later in the question generation phase.

Let us have a look at the following sentence (from our initial RoboCHAIR paper (Pollak et al., 2015)): "In a similar way, we used ConceptNet to find theme words by inspecting all the IsA relations in its database, from which it identified 11,000 themes.". The catch phrase is formed by the pronoun `we` and the verb `use` in the VBD (Verb, past tense) form. The sentence also does not match the criterion for stopping: the catch phrase is followed by a noun phrase `ConceptNet` and not by a template-specific stop word `to` (to exclude the phrases of a type `used to`). In this sentence, `ConceptNet` corresponds to `object X`, which is used in the question generation phase.

**Question generation** For each category, a corresponding template is activated. For the example above, the template is "What if you $VBD something else instead of $X?"

Variable $VBD$ is replaced by a past-tense form of the verb from the formula that was used when finding a pattern match (in our case `used`), and variable $X$ is replaced by the whole object X, as extracted from the candidate sentence parse tree in the selection phrase (in our case `ConceptNet`). If object X exists in a domain specific ontology, its hypernym is used instead. Instead of a single template, several templates are proposed to improve the diversity for each pattern.

Below, we provide few examples from the evaluation presented in this study, together with the linguistic category of the anchor. We list the input sentence form the sentence selection phase, and the resulting generated question.

- **Divide**
  Sentence: *For sequences longer than 512 tokens after tokenization, we took 256 tokens from the beginning of the text and 256 tokens from the end of the text and concatenated them together.*
  Question: *Why did you decide to take 256 tokens from the beginning of the text and 256 tokens from the end of the text and concatenated them together?*

- **Focus**
  Sentence: *We focus on the hate speech recognition task.*
  Question: *Why did you decide to focus on the hate speech recognition task?*

- **Usage**
  Sentence: *We use sentence-level rewards to optimize the extractor while keeping our ML trained abstractor decoder [fi]xed, so as to achieve the best of both worlds.*
  Question: *Do you think you could use something else instead of sentence-level rewards?*

Other examples of sentences can be seen in Figure 1. The first eight sentences, generated based on the input paper by Pranjić et al. (2020), are presented (the file was uploaded as a .tex file). The first six questions on the picture are the result of the pattern-based method. We can see that the category usage was the source of the majority of the questions on the picture.

**A triplet graph embedding method (TEM)**

In order to overcome the limitations of the pattern-based sentence selection we have developed an advanced approach which tries to mimic human understanding and sentence selection for question generation. The process of question generation in humans is based on deep understanding of the text while adding all the available background knowledge of the reader. The content that cannot be sufficiently explained using this procedure requires additional information which can be obtained by formulating questions about the relevant parts of the input.

Our approach approximates this process by extracting the essence of the text in the form of a triplet graph which is followed by inserting additional edges between triplet parts using similarity queries based on word embeddings. Our assumption is that in the end the nodes with very low number of outgoing edges are the ones that require additional explanation because they are not sufficiently interlinked in the triplet knowledge graph. In comparison with human question generation, triplet extraction mimics extracting the content while word embeddings plays the role of the background knowledge database which connects entities into a coherent picture. In the following we provide a detailed description of the proposed method.

Triplet extraction is a common way for information extraction from unstructured text data. A triplet consists of subject, predicate and object and defines a binary relationship between the subject and the object. Given a chunk of text one can extract triplets and construct a triplet graph which is a visual summary of the text. On the other hand, word embedding enables the mapping from words to vectors of real numbers which allows for various computations such as distance (similarity). Using word embedding, "nearest neighbours" of words can be easily obtained.

The TEM method combines the two methods by taking the triplet graph and computing additional edges using word embedding. Taking into account that word embeddings can be trained on specific domains, this approach mimics the use of human background knowledge during the cognitive process when the recognised entities are grounded. Given some input text, a trained word embedding and a triplet extractor, the algorithm works as follows:

1. extract triplets from the document;
2. create a triplet graph (a directed multigraph) by adding subjects and objects as nodes and predicates as directed edges connecting subjects with objects;
the source sentence selection is composed of a list of linguistic anchors and a database of catchwords. The questions will be used for generating the article.

Figure 2: A triplet graph with several additional word embedding based edges. Edges between triplet parts are in black while word embedding edges are in gray. Terminal nodes without outgoing edges are shown in red colour and represent good candidates for question generation.

3. for each node, find \( n \) nearest neighbours\(^3\)
4. scan all nodes and insert directed edges if a nearest neighbour of some node is contained in another node;
5. the result is a set of nodes
   \[ A = \{a \mid m \leq \deg(a) \leq M\} \]
   for some user-defined thresholds \( m \) and \( M \).

The algorithm return a set of nodes (triplet subjects and objects) which corresponding source sentences are candidates for question generation. There are three parameters which affect the size and quality of the result:

1. number of nearest neighbours: \( n \)
2. lower bound for node outdegree: \( m \)
3. upper bound for node outdegree: \( M \)

The number of nearest neighbours is related to the size of the triplet graph. When the graph is large, the number of neighbours should also be large. The key observation when adjusting \( n \) is that if it is too high, the graph will be overconnected and no node will have a low outdegree. On the other hand, if the number of neighbours is too low, unconnected components may appear and give a false impression of importance of the corresponding nodes. As a rule of thumb, the number of neighbours is proportional to the size of the input until some upper limit. For example, for a text consisting of one or two paragraphs, \( n \in [1, \ldots, 10] \) is sufficient, while for a text consisting of several pages \( n \in [100, \ldots, 1000] \) is an appropriate choice.

The lower and upper bound parameters \( m \) and \( M \) are only used during the selection of the result set and have no effect on graph construction. In general, when the number of neighbours is within reasonable limits, the nodes with zero or one outdegree are true outliers. For example, misspellings, names, abbreviation, foreign words etc. are often found in such nodes. Therefore, it is recommended to set \( m \) to some low number and increase it only if true outliers still appear in the result. The upper bound \( M \) is used to limit the number of results but can be determined automatically by gradually increasing it, starting with \( m + 1 \). This way, it is possible to return the desired number of results.

We illustrate the TEM method in the following example.

**Example.** Suppose we have the following text (an excerpt from our initial paper on conference management assistant (Pollak et al., 2015)) for which we want to identify parts that are considered relevant for asking questions:
Question generation module
Our question generation system (depicted in Figure 1) is composed of the source sentence selection, i.e. detection of sentences in the article that will be used for generating the questions, and question formulation. The system’s input are preprocessed text documents, which are uploaded and converted into raw text on the online platform (see Section 3).

Pattern-Based Selection of Source Sentences
We begin with a list of linguistic anchors, i.e. a database of catchwords that enables us to select candidate sentences as a source for question generation. The sentence matching process has two phases: the coarse-grained and the fine-grained sentence selection process. We decided to use relatively strong conditions for selecting candidate sentences, since we believe that - in order to achieve higher quality - it is better to miss some good candidates in the process of selection than to generate too many non-relevant questions.

Using the ReVerb triplet extractor (Fader, Soderland, and Etzioni, 2011) and GloVe word embeddings (Pennington, Socher, and Manning, 2014) we obtain a graph which is shown in Figure 2. The graph was constructed by representing each triplet <subject, predicate, object> as a set of nodes connected with a directed, labelled edge. In addition, 5 nearest neighbours of each node obtained by the GloVe embeddings trained on Wikipedia were considered for adding additional edges. The lower bound threshold $m$ was set to 0 and the number of desired results was set to 3 (this turned on the auto tuning of the upper bound threshold $M$). The nodes in the result set were labelled as terminal nodes and coloured in red. We conclude that the phrases or terms in terminal nodes are not sufficiently explained in the text and should be considered while formulating questions. For example, the graph in Figure 2 can identify the following nodes, which could inspire the following questions:

- **two phase**: Please explain the two phases of the sentence matching process.
- **some good candidates**: Can you give some examples of good candidates?
- **many non-relevant questions**: How do the generated non-relevant questions look like in general?

The questions are highly relevant to the corresponding text and clearly demonstrate that the proposed method is able to identify weak points in its input. However, in the current implementation, the generation uses a simple generator template “Can you elaborate on X”, where X is the identified triplet graph node.

In Figure 1, the last two sentences are the output of the TEM module.

Evaluation
The two methods were evaluated by six computer science students, who evaluated 11 papers in total. They were asked to select two papers, one that they authored and one that they read. The first setting corresponds to the *Author assistant mode* and the second one to the *Conference chair assistant* functionality. When uploading the papers, they were asked to use the default setting (both generation methods) and not only the faster pattern-based generation option. They were also asked to indicate whether they are evaluating the questions for the article they wrote (system used as *Author assistant*), or if they evaluate the paper that they did not author, (system used as *Reviewer assistant*).

The evaluation criteria were the same as in Pollak et al. (2015):

- **Understandability/Acceptability** is a binary category verifying if the question was understandably formulated. The evaluators were asked not to penalize smaller mistakes (grammatical or PDF conversion errors), but to give negative answers if the question is not understandable.

- **Meaningfulness/Relevance** is scored on a scale from 1 star (irrelevant) to 5 stars (very relevant), with the following description: $5=very$ relevant(meaningful, related to the topic, no semantic issues), $4=relevant$ (meaningful, minor semantic issues), $3=partly$ relevant (good but partly impertinent, some semantic issues), $2=not$ relevant (too trivial, big semantic issues), $1=completely$ irrelevant (not meaningful, wrong).

Regarding the selection of evaluation categories, our work was inspired by previous studies: the binary score evaluating if the question was acceptable (i.e. understandably formulated) can be related to the “acceptable vs. unacceptable” binary scores in Liu, Calvo, and Rus (2014); Chali and Hasan (2012). Next, our 5 star meaningfulness/relevance score can be aligned with the evaluation of (topic) relevance in (Chali and Hasan, 2012) but adapted, as in their study factual questions were generated.

The evaluation was focused on comparing the two question generation methods: the legacy pattern-based method which was already found to produce moderately relevant and understandably formulated questions and the new method based on embeddings and triplet graphs which employs a mechanism to identify under-explained parts but currently uses only one general template to construct questions.

In total, for the 11 scientific papers, the system generated 306 questions. 5.3% of the pattern-based approach and 22.2% of TEM-based were not understandable. In total, 273 questions were rated as “understandable”. Taking into account only those, the pattern-based system generated 196 questions where triplet embedding method (TEM) generated 77 questions.

The uploaded articles contained in average 624.18 sentences (the shortes article had 254 sentences and the longest
one 1572). The mean number of generated questions per paper was 24.8 and the standard deviation 14.6. The pattern-based method does not control the number of questions, as the number of resulting sentences depends on the sentence matching step. In contrast, TEM can return a desired number of questions. In order to balance the number of questions generated by both methods we configured TEM to match the number of questions produced by the pattern-based method if the number of pattern-based questions was between 5 and 10. Outside of this range the lower limit was set to 5 and the upper limit to 10.

The mean meaningfulness score for all understandable questions was 3.03, while if considering only the pattern-based method, the score was 3.17 and the mean score of the TEM method was 2.68. Compared to the initial pattern-based method from the first RoboCHAIR version, the score for the pattern-based approach increased from 2.99 to 3.17, and the overall score (including both methods) to 3.03. The TEM method average score (2.68) is also above the threshold of 2.5 which was selected for keeping the pattern-based question templates in the system. The distribution of the meaningfulness score for both methods is shown in Figure 3.

![Figure 3: The distribution of the meaningfulness score of understandable questions for both question generation methods.](image)

The TEM method received many lowest possible scores. An investigation revealed that the generated questions were ranked low because of the outliers that were identified as interesting nodes. This suggests that the default lower bound $m = 2$ is too low and the outlier nodes in the TEM graph are mistakenly identified as relevant. For example, names of people, methods, abbreviations, numbers, equation parts, etc., are targeted for question generation. This problem can be almost perfectly resolved by increasing the lower bound, improving text conversion, filtering triplets and using domain-specific embedding model.

Takings into account the authorship of papers the meaningfulness score reveals a bias which is present in both methods but especially notable in the pattern-based method (see Figure 4). For evaluators who were not the authors the peak is close to 4 while for authors the peak is close to 3 which suggests that authors which have a detailed knowledge about their papers consider questions less relevant or possibly too simple.

While in the current implementation the TEM method is outperformed by the pattern-based one, we believe that the TEM method still has a lot of potential for improvement. Currently, the method focuses only on the sentence selection phase, but not on the question generation one, which is in our opinion one of the reasons of the lower results. This might be the source of a positive bias towards questions generated with the pattern-based method which uses more elaborate templates to generate questions. In few cases, both methods selected the same sentence as relevant and formulated a question. However, the question generated using the templates from the pattern-based approach was evaluated much higher than the question generated with the TEM method, which uses one general template. For example, based on the same input sentence, both methods generated the questions, which were ranked as 4 for the pattern-based method and 1 for TEM:

**Sentence:** We took 1,430 tweets labeled as the hate speech and randomly sampled 3,670 tweets from the remaining 23,353 tweets.

**Pattern-based question (score: 4):** Why did you decide to take 1,430 tweets labeled as the hate speech and randomly sampled 3,670 tweets from the remaining 23,353 tweets?

**TEM question (score: 1):** In the previous sentence you mention 1,430 tweets. Can you please elaborate on that?

This indicates that more elaborate generation part following the sentence selection by TEM could improve the results.

The number of overlapping sentences selected by the two methods is very low and mostly coincidental because the pattern-based method selects sentences according to predefined patterns which are suitable for question generation while TEM selects sentences according to the connectivity of nodes in the triplet-embedding graph.

In one of the comments, the evaluator also explains that while in an application oriented paper, the system achieved
quite relevant questions, in an overview paper that he uploaded, the questions were not that relevant.

Conclusions and further Work

This paper describes the updated RoboCHAIR system. The paper was evaluated by the students. In contrast to the main RoboCHAIR functionality, where the system was designed to help conference chairs, the students are also one of the core target groups, and the system aims to support them in the process of scientific writing of papers or reviews. The questions are designed to model human intelligence by triggering new scientific ideas or making the authors explain the decisions behind their approach, using two different methods, one based on patterns and one on triplet graphs. The first method was evaluated with higher scores, but we believe that as the generation module was more developed in the pattern-based approach, the triplet graph based method could be further improved in the future.

For example, in TEM method multi-word phrases during neighbourhood search are currently decomposed into words and the corresponding vectors are added to get the final vector. A possible improvement would be to consider bigger units, e.g., named entities instead of single words. Yet another interesting addition would be to use sense embedding (Camacho-Collados and Pilehvar, 2018) instead of word embedding to account for different meanings of words, or consider mapping contextual embeddings to static graph nodes. Finally, the discovered nodes with low outdegrees contain only parts of sentences and the problem of formulating the actual question about the under-explained part uses very simple templates. In future the question generation part of TEM should be further improved.

In terms of evaluation, in future work it would be interesting to compare automatically generated questions to question generated by humans. Currently, the evaluation criteria are relatively general, and in future it would be interesting to introduce scores explicitly focused on the novelty of the questions, as well as to get feedback more specific to the actual use. Last but not least, using the evaluations in an active learning model to identify relevant would be a valuable addition.

Acknowledgements

We acknowledge the project the financial support from the Slovenian Research Agency for research core funding for the programme Knowledge Technologies (No. P2-0103).

References


Abstract

Although creativity is an ability that can be developed if trained, most of the systems developed for creativity stimulation focused on individual interventions. This work presents the design, development, and evaluation of CUBUS, a digital tool developed to inspire social creativity in children through storytelling and emotionally expressive characters. Groups of children collaborate with the autonomous virtual characters of the digital tool to create a story together. Our main goal was to conduct an experimental study (n=20) to investigate if the interaction between children and the autonomous emotionally expressive virtual characters present in CUBUS can stimulate social creativity during a storytelling activity. We measured the impact of the digital tool across several creativity domains: pre-and post-testing (questionnaire assessment), the creative process of children (behavior analysis), and the story created (creative outcome). Results showed that although children generated fewer storytelling ideas using CUBUS, these ideas were deemed more original.

Introduction

Despite the major role of creativity in our lives, including being paramount for our professional success and well-being, it is often overlooked and unappreciated. Although some schools feature activities that foster creativity, such as theater or painting, they appear as scarce and often times are not deemed mandatory. The mere time for playing, a precursor time for imagination and creative thought, is being reduced from children’s lives (Elkind, 2008). This shows how children’s curiosity and creativity tend to be dismissed starting in early school years (Kim, 2011). The formal organization of schools (e.g., reinforcing behaviors such as staying seated for extended periods and performing large amounts of school homework), seems to be driving the creative growth of children outside of the schools more than inside its walls (Csikszentmihalyi, 1996; Runco, Acar, and Cayirdag, 2017). It becomes thus necessary to find new ways to honor creative spaces at school.

An important aspect of fostering creativity at school is to endow teachers with the necessary tools and support to accomplish this task. Although many schools already feature storytelling activities that help promote children’s creative thinking (Di Blas, Paolini, and Sabiescu, 2010), these activities are cumbersome for teachers to prepare and manage, not to mention they are scarce (Chan and Yuen, 2014). With this work, we aim to contribute to current creativity support tools with the design of CUBUS, which consists of a digital environment platform with emotionally expressive characters for storytelling activities. CUBUS can be used in any compatible tablet or iPad, as children seem to show an interest in using electronic devices from a young age (Salonius-Pasternak and Gelfond, 2005). The novelty of CUBUS is the presence of emotionally expressive autonomous virtual characters that hint at new storylines that would not emerge otherwise. To develop this digital tool, we included children as design partners, testers, and users who lead the design, improvement, and evaluation of this tool in terms of its impact on their creativity.

Our ultimate goal was to investigate if the interaction between children and the autonomous emotionally expressive virtual characters present in the digital tool can stimulate social creativity during a storytelling activity. CUBUS shows promises to be used in education environments, such as the school, and for any collaborative play environment of children, such as their home.

Background

This section provides a background on the topics of social creativity and the role of emotions in creativity. We reviewed these concepts as they informed the design and development of our digital tool.

Social Creativity

Researchers have different viewpoints as to what creativity is (Kampylis and Valtanen, 2010) with the agreement over
this concept changing over time (Sternberg and Sternberg, 1999). While earliest definitions of creativity described this ability as a function of an individual (Guilford, 1967), creativity has been defined considering more than just the isolated individual nature and expertise to be seen as a social construct (Plucker, Beghetto, and Dow, 2004). The focus of this work will be on social creativity, also described as distributed creativity (Sawyer and DeZutter, 2009). This particular aspect of creativity is related with solving problems and creating new solutions collectively. Social creativity is crucial to be studied, since “much of the human creativity arises from activities that take place in a social context in which interactions with other people and the artifacts that embody group knowledge are important contributors to the process” (Fischer et al., 2005). This corroborates that creativity does not occur strictly within each individual but also through our interactions with each other when pursuing a common focus (Csikszentmihalyi, 1996).

As human beings are social by nature, it is therefore expected that a significant part of our thinking and problem solving emerged collaboratively (Fischer et al., 2005). For social creativity to unfold, there are some requirements that need to be met, such as the degree of task complexity which should account for some uncertainty as a way of generating discussions and collaborative solutions; it is equally important that the task is unscripted and that it allows for experimentation, so that improvisation and flow come into the creative process (Fischer, 2007). In our work, we relied on these principles and chose a storytelling task as it denotes some structure, accounting for the existence of characters, action, and a scenario, and is anchored on a starting, middle, and ending plot stages. At the same time, the open-ended nature of a story provides children with the right amount of uncertainty to create storylines together.

**Emotions in Creativity**

According to Hutton and Sundar (2010), games created with the goal of increasing creativity have much to gain from featuring interactions that are rich in emotional expression. Different levels of emotional arousal impact creativity. While higher levels of arousal tend to hinder the creative process as they reduce the capability to perceive, process, and evaluate new information (St-Louis and Vallerand, 2015), moderate levels are helpful when gathering information to create novel ideas since they enhance cognitive flexibility and help to stay focused during the task. Additionally, a “boomerang effect” of arousal is found to be detrimental in divergent thinking tasks, as moderate levels of arousal (rather than extremely low or high levels) are necessary for engagement in divergent thinking performance (De Dreu, Baas, and Nijsstad, 2008).

As emotional expression provokes creativity, we have incorporated the idea of emotionally expressive virtual agents in our digital tool. These characters favor moderate emotional arousal levels that are reported as beneficial to stimulate creativity, rather than extreme emotional reactions.

**Digital Tools for Creativity**

There are a few digital platforms developed that closely relate to creativity applications. Hornecker and Stifter (2006) developed tangible interfaces to engage visitors in a museum experience. Their main findings indicated that engagement is fostered between the museum audience and the tangible experience if the system can accommodate group interactions, as most museum visitors come in groups. Additionally, Snibbe and Raffle (2009) created guidelines for designing users’ interactions with social immersive media, which include narrative models that highlight the power of ‘users as actors’ to promote natural and engaging interactions. Additional work on prolonged engagement with digital media and museums exhibits was conducted (e.g., Humphrey and Gutwill (2017)), and a common theme is to place the users as the active (and not passive) part of the experience. The incorporation of storytelling in digital interactive systems showed that such systems can create a medium for collaborative expressive interaction between the users and the system itself, especially when the system can convey emotions by making use of colors and animations (Long et al., 2017), similarly to what we incorporated in this work.

**CUBUS: Our Digital Tool for Creativity**

In this section, we provide details about CUBUS, the digital tool that we created1. CUBUS consists of a virtual environment that can be accessed using an iPad or Tablet and enables children to build stories using cube-shared characters. These virtual characters are designed to be emotionally expressive and CUBUS is intended to foster social storytelling experiences for children. In this section, we provide details about the design of the virtual characters and the virtual world of CUBUS.

**Emotionally Expressive Characters for Creativity**

The virtual agents that act as autonomous characters in the story are one of the most innovative parts of this work as they were designed to be the stimuli that would foster children’s creative process during storytelling. To avoid the uncanny valley effect (Ciechanowski et al., 2019), especially concerning artificial emotion expression, the design of these characters’ space revolved around non-humanlike characters (Tinwell et al., 2011). Therefore, the characters are

---

1A tutorial video about CUBUS – our digital tool can be accessed through this weblink: https://www.youtube.com/watch?v=Tuj5-27fqwY.
shaped in the geometric form of a cube. This format of characters has several advantages, namely avoiding stereotyped ideas about gender, role, and behaviors and leaving the storytelling more open to children’s ideas. Additionally, it is well known from previous research that humans can create complex stories with abstract shapes, such as triangles and rectangles, which supports our design choice (Heider and Simmel, 1944).

The characters make use of minimal interaction modalities, such as colors and movements to express emotional states. It is well established the relation between color (Gilbert, Fridlund, and Lucchina, 2016; Sutton and Altarriba, 2016) and movement (Wallbott, 1998; Camurri, Lagerlöf, and Volpe, 2003) in emotions. The colors for each emotion expressed by the characters followed studies about color-emotion association models (Terwogt and Hoeksma, 1995; Hemphill, 1996; Nijdam, 2009; Terada, Yamauchi, and Ito, 2012; Baraka, 2016) and were inspired in film-making through the Pixar® animation movie “Inside Out”. We have chosen this movie as an inspiration as Paul Ekman was the scientific consultant for the design and creation of the movie characters’ whose role is to represent emotions (Keltner and Ekman, 2015). Ekman’s insights for color-emotion-behavior mapping for Inside Out overlaps with previously referred models, e.g., similarly to Terada, Yamauchi, and Ito (2012)’s model, anger is represented as red and joy as yellow, providing scientific ground to this source of inspiration. As such, we considered that the unique colors and behaviors associated with each emotion would create distinguishable and appropriate emotions for each of our characters (see Figure 2).

The virtual agents’ design followed Disney® animation principles to provide the “illusion of life” of the characters (Thomas, Johnston, and Thomas, 1995). From the twelve established animation principles, we have chosen four that could be transferred to our characters. These are squash and stretch (characters squash and stretch deforming their initial form while maintaining the same volume), anticipation (anticipating movements and actions that inform what the character is going to do next), follow-through (this principle works as an opposite of Anticipation. When a character stops doing something, it should not stop abruptly, for that causes an unnatural feeling; this animation is generally associated with inertia but can also be used to emphasize the stop), and staging (related to the general set-up in which the character expresses itself; this principle is related to making sure that the expressive intention is clear to the viewer. Some ways of accomplishing this are by positioning lights, camera, music, characters, and surrounding objects). We used the Autodesk Maya®, a 3D animation software application, to create animations for the virtual agents.

Since emotions tend to enhance creativity in video games (Hutton and Sundar, 2010) it was important to establish the most appropriate emotions to model the characters. We used Ekman’s model of basic emotions and included happiness, anger, sadness, fear, and disgust to develop in the characters (Ekman, Friesen, and Ellsworth, 2013). Surprise emotion was removed given its ambiguous interpretation in terms of valence and associated age-differences in its perception (Tottenham et al., 2013; Jack, Garrod, and Schyns, 2014; Shuster, Mikels, and Camras, 2017). Each emotion has a unique appearance and means of expression, detailed below (see also Figure 1).

Happiness — Characters featuring happiness express fast and expansive movements, such as jumping or dancing. These animations resemble positive and playful actions, such as spinning jump, inspired in a celebration. The color of happiness is yellow;

Anger — Characters featuring anger convey an aggressive movement. They stretch their bodies and mimic an inflated chest while leaning forward and keeping the tension in the character’s body. The color of anger is red;

Sadness — Characters featuring sadness express slow and contained movements, such as appearing contracted with their “head” hanging low giving the impression of being looking at the floor. This character can whimper to mimic crying. The color of sadness is blue;

Fear — Characters featuring fear quickly retract, crouch, and hide. While crouching, the characters start twitching and shaking in fear. The color of fear is pink;

Disgust — Characters featuring disgust appear as looking away and avoiding contact. This effect was emulated by having the character retracting and turning away, giving the impression that “it cannot look to a disgusting stimulus” before reluctantly turning back to its original direction. The color of disgust is green.

Characters express their emotion more intensively according to the distance they have from each other (what is called 2The design and behavioral expression of each of the characters’ emotions is also detailed in a video that can be accessed through this weblink: https://www.youtube.com/watch?v=oDAm__9eyjw.)
proxemics) (Hall, 1966). This parametrization was developed in order to create the behavior of the characters more complex in terms of their emotional expression, i.e., a character that is closer to another has a stronger emotional expression and the degree of expression decreases as characters are further away. This behavior was designed to mimic the lower/higher perception of emotions with distance (see Figure 3 on the right for a representation of proxemics). While using CUBUS children can add, remove, or hide characters for their story. This can be performed using a dedicated menu on the interface. The characters can be moved around the virtual world using drag-and-drop, a traditional feature that exists in most digital devices.

**Virtual World for Storytelling**

The virtual world of the digital tool supports children’s creative storytelling process through the use of components such as background personalization, stop-motion technique, and intertitles. Specific features of the world are detailed below.

**Backgrounds** — Background’s shapes and colors can be customized according to the story children want to create. Background can vary from curvilinear, rectilinear, or spiky. A color palette was enabled, and the background can be adapted to any color (e.g., to simulate the different phases of a day, weather conditions, or environments) (Ware, 2012) (see Figure 4);

**Stop-motion** — The digital tool supports the stop-motion technique for storytelling as children can record the story they create. While they create the story, children perform screen captures that are stored in the digital tool. In a later stage, these frames are transformed into a movie that allows for narration (see Figure 3);

**Intertitles** — Intertitle screens can be incorporated at any moment of the story, similarly to their usage in silent movies (see Figure 4).

**Experimental Study**

This section presents the experimental evaluation to investigate the impact of the digital tool in stimulating creativity in children. Therefore, the research question for this study was: *can the interaction between children and the autonomous emotionally expressive virtual characters present in the digital tool, stimulate social creativity during a storytelling activity?* To answer our research question, two study conditions were considered:

- **Experimental condition**: Small groups of children created a story using the digital tool, featuring autonomous and emotionally expressive virtual characters;
- **Control condition**: Small groups of children created a story using CUBUS, but its characters did not display any behaviors.

Given that emotions positively influence creativity expression (Hutton and Sundar, 2010), we hypothesize that children in the experimental condition will score higher in creativity levels compared to children in the control. Children’s creativity levels were measured in terms of their *creative process* during story creation (measured through behavior analysis using the recordings of the sessions), their *creative product* (the final story created by children was evaluated with external judges experts in the field), and the impact in their creativity skills, often called *creative person* in creativity research (measured with a creativity validated test as a pre- and post-test evaluation). By measuring the impact of CUBUS across these different domains of creativity, a deeper understanding of the impact of interacting with emotionally expressive autonomous agents in creativity is acquired.

**Participants**

A total of 20 children participated in the final evaluation of the system. Children ages ranged from 7 to 9 years old ($M = 8.10$, $SD = 0.72$, 14 female). This study was performed in the classroom of a school and children performed the task in pairs chosen by the school teacher, therefore, each session consisted of 2 children in a total number of 10 sessions. Each session lasted approximately one 1-1:30hours. The difference in the length of the session is attributed to the time that children took while creating the story, which was not restricted.

**Materials**

CUBUS was used to perform the storytelling activity and run on an Android tablet. We also used voice recordings of children to collect data about the creative process of storytelling.

**Measures**

In this section, we detail the measures used to evaluate the creative person (creative skills of children), creative process (final story movies), and the creative process (story creation).

**Creative Person.** We used the Test for Creative Thinking-Drawing Production (TCT-DP) test to measure the creative potential of children. TCT-DP is a well-established test in the field of creativity, applicable to a broad age range, culturally fair, and helps to identify high creative potentials as well as low creative, neglected, and poorly developed ones (Jellen and Urban, 1986; Urban, 2005; Jellen and Urban, 1989). A version of the test adapted to the Portuguese
population was used (Nogueira, Almeida, and Lima, 2017).
TCT-DP is composed of Forms A and B and consists of a
sheet of paper with six graphic elements, named fragments,
of a circle, a dot, a dashed line, a 90-degree angle, a curved
line, and a small open square. These are placed at fixed
and pre-established locations on the page. All of the ele-
ments, except for the small open square, are enclosed in a
large rectangular frame, and this forms a short of an incom-
plete drawing. According to the manual, participants are in-
structed to “complete the drawing initiated by an artist” and
to “give a title to the drawing when completed”. The final
drawings made by children were scored according to the 14-
point scoring system (Urban, 2005) and a trained psycholo-
gist that underwent TCT-DP training scored each drawing.
The specific criteria used to code the drawings and the de-
tailed application instructions can be found in the Supple-
mentary Materials of this submission.

Creative Process. The creative process of children was
evaluated using a deductive content analysis approach and
inter-rater reliability was established and a coding scheme
was developed. To do this, two psychologists blinded to
the hypothesis of the study analyzed the video recordings
of children creating a story and established categories. Dis-
agreements were settled by several joint discussions and ad-
justments to the criteria for each category. The final coding
scheme was used to analyze the interactions and the ideas
generated during the creative process. This coding included
three broad categories related to creativity: fluency, flexi-
bility and elaboration, and originality. Agreement rates
were calculated separately for each dimension of the cod-
ing scheme. Overall, the inter-coder agreement was high,
varying between 0.76 and 0.95 (M = 0.87), indicating an ex-
cellent level of agreement according to statistical standards
(Bakeman and Quera, 2011).

Creative Product. The final short stop-motion movies
created by children were evaluated using the Consensual As-
Sessment Technique (CAT), which relies on ratings given
by a panel of independent expert judges (Hennessy, Am-
able, and Mueller, 1999). In the case of this study, the
panel of external judges consisted of experts of cinema and
animation ranging from movie directors, producers, anima-
tors to cinema college teachers. We identified these judges
by performing an online search using Google Search En-
gine for local cinema-related activities, using the keywords
of cinema festivals for children, animation festivals, cinema
schools, and cinema universities. We then established con-
tact via email to understand their availability to participate
as judges and, upon their agreement, each judge rated 10
movies, 5 from each test condition. Their coding was indi-
vidual and blinded to the study condition. To eliminate or-
der effects for movies’ assessment, we used the Latin square
technique (Winer, 1962) to randomize the questionnaires’
version when the judges opened the link provided. This
evaluation was performed through an online questionnaire

3Details about the coding scheme are present in this
weblink: https://osf.io/6cv93/?view_only=2c002947ad8046a0a5ee4f5a9.

Figure 5: Children interacting with CUBUS, the digital tool
for creativity inspiration.

Using Google Forms containing children’s movies accompa-
nied by a scale which allowed rating them, as dictated by
CAT. Our final panel of judges, i.e., judges that finalized the
evaluation of all movies, was composed of 9 participants,
aged between 19 – 54 years old (M = 42.00, SD = 10.10, 7
male). Given that the assessment is performed with experts
in the field, small samples such as 5 – 10 experts is accepted
in the literature (Hennessy, Amabile, and Mueller, 1999).

Procedure
Pairs of children entered the designated classroom where
the study was performed (see Figure 5). The groups of
children were chosen by the teacher. Two researchers (one
psychologist and one computer scientists) were involved in
conducting this study. Children started to create their story
collaboratively by adding characters and personalizing their
scenario/background. The last stage of this study consisted
of watching the movie with the children and congratulating
them on their accomplishment.

Results
This section details the results regarding the creative person,
process, and product.

Creative Person
Shapiro-Wilk test for normality revealed a normal distribu-
tion for TCT-DP Form A, p > 0.05, and a non-normal distribu-
tion for Form B, p < 0.05. To assess if the sample
was homogeneous in terms of creativity skills before the
intervention as a baseline measure, we computed the
results of the pre-test (Form A) between conditions, and
no significant differences between creativity skills in chil-
dren allocated in the experimental and control conditions
t(18) = 0.872, p = 0.394, revealing a similar creative po-
tential of our participants. To measure if the intervention
had an effect on the creativity skills of children we com-
pared the results from the post-test (Form B) and concluded
that there is a borderline significant effect between condi-
tions, U = 24.500, p = 0.053, r = 0.4, with children

0 10 20 30 40 50

153
showing higher creative skills in the experimental condition \((Mdn = 13.05)\) compared to the control condition \((Mdn = 7.95)\). Overall, this result did not support the hypothesis which postulated that creative levels of children would be higher in the experimental condition.

Creative Process

The story creation process lasted, in average, 30 minutes \((M = 29.67, SD = 8.33, min = 21.35, max = 47.18)\) and resulted in the generation of an average of 174 ideas per session \((M = 174.40, SD = 68.44, min = 100, max = 333)\). To evaluate the effects of the manipulation in the children’s creative process, we performed a between-groups analysis by calculating the rate of ideas generated per category of the coding scheme considering different storytelling stages4. In particular, we used three continuous but mutually exclusive storytelling stages, namely rising action (starts at the beginning of the interaction and ends when the characters in the story enter the main action), climax (from the ending of falling action until the moment when the characters finish the main set of actions), and falling action (comprises interactions occurring afterward until the end of the story) (Freytag, 1896). To calculate these rates and to accommodate for the group nature of the interaction, we considered the number of ideas produced by the group (and not the individual) as a unit of measure of social creativity. To analyze the difference in the rates of ideas belonging to each storytelling stage, we conducted an independent samples t-test in which we compared the ratio of ideas in each story-building stage between the two conditions.

During the climax stage, we found a difference in fluency between conditions, \(t(8) = −3.23, p = 0.01, d = 2.01\), with the control condition showing a higher rate of relevant ideas generated \((M = 0.37, SD = 0.04)\) compared to the experimental condition \((M = 0.23, SD = 0.09)\). We did not find additional results for rising action and falling action between conditions. In addition, we observed significant differences between conditions in the fluency regarding ideas generated for the characters, \(t(8) = −4.01, p = .004, d = 2.75\), with participants in the control condition generating more relevant ideas \((M = 0.12, SD = 0.03)\) than in the experimental condition \((M = 0.05, SD = 0.02)\) (see Figure 6).

We then performed a within-conditions analysis by comparing the fluency for each storytelling stage (rising action, climax, and falling action) using a paired samples t-test. We found a statistically significant difference in fluency between the rising action and the climax stages, \(t(9) = −8.88, p < 0.001, d = 2.32\). This suggests that children generated more relevant ideas during climax \((M = 0.30, SD = 0.10)\) than during the rising action stage \((M = 0.09, SD = 0.08)\). Specifically, a significant difference in the fluency related to the scenario was observed between the rising action and climax stages, \(t(9) = −2.38, p = 0.04, d = 1.13\), suggesting more ideas related to the scenario during the climax.

\footnote{The coding scheme used to code the creative process can be accessed using the weblink: https://osf.io/6cv93/?view_only=2c002947ad8046a0afa54b7a5ee4f5a9.}

max \((M = 0.08, SD = 0.04)\), than during the rising action \((M = 0.04, SD = 0.03)\). A similar pattern was observed in the fluency related to story actions. Namely, a difference between the rate of ideas generated during the rising action \((M = 0.03, SD = 0.004)\) and the climax stages \((M = 0.13, SD = 0.07)\) was found, \(t(9) = −5.51, p < 0.001, d = 1.75\). Results also showed a significant difference between the rate of fluency towards the character during rising action and climax stages, \(t(9) = −5.00, p = 0.001, d = 2.21\); and during the falling action and the climax stages, \(t(9) = −2.53, p = 0.03, d = 1.10\). Specifically, children generated a higher rate of ideas related to the character during climax \((M = 0.09, SD = 0.04)\) than during the rising action \((M = 0.02, SD = 0.02)\) and falling action \((M = 0.04, SD = 0.05)\). These results showed that independently of the group, and during the climax stage, children generated more ideas for their stories related with the scenario, action, and characters (see Figure 7).

Regarding the elaboration of story actions, we observed a significant difference in the rate of ideas generated during climax and falling action stages, \(t(9) = −2.69, p = 0.025, d = 1.00\), and between the rising action and climax stages, \(t(9) = −3.65, p = 0.005\), favoring elaborations during climax. A similar pattern was found for the elaboration of the scenario, in which a statistically significant difference was found regarding elaborations about the environment where the characters are in the story space, deemed higher during the climax, \(t(9) = −3.27, p = 0.010\). We also found a significant difference on elaboration regarding character dynamics during rising action and climax, \(t(9) = −4.10, p = 0.003, d = 1.00\), favoring the climax stage. This suggests that children added/removed/hid characters in the peak of the action the story (climax), as some characters are present only in certain parts of the story and reappear later on. This result is also consistent with previous results that highlight that children’s creativity peak during the storytelling is more emergent during the climax, which is the central moment of the story.

The last set of analyses performed concerns the originality of the ideas created regarding the characters, action, and scenario. To analyze the originality, two external coders used a Likert scale of 7 points in which originality was defined

Figure 6: Results from the experimental study regarding the creative process of storytelling between conditions. On the left: Control condition showed higher fluency levels than the experimental condition, \(p < 0.05\). On the right: Experimental condition showed a higher originality level of the creative process compared to control, \(p < 0.05\).
as the “uniqueness (rarity) of an idea in relation to a set of ideas” (Shamay-Tsoory et al., 2011) to code each idea generated during the creative process. For this, voice recordings were transcribed, and the coders coded each child’s utterance in terms of originality. We observed 0.97 of agreement regarding the character category, 0.98 for the scenario, and 0.97 for action. Furthermore, we conducted t-tests to analyze the differences in the originality of ideas for each category and found significant differences in the originality of ideas only related to the character, t(8) = 2.65, p = 0.03, d = 1.66. These results suggest that more original ideas towards characters were generated in the experimental condition (M = 4.60, SD = 2.10) compared to control (M = 2.00, SD = 0.71) (see Figure 6).

Overall, the results show that the number of ideas generated (fluency) was superior in the control compared to the experimental condition. However, the originality of ideas regarding details about the characters was superior in the experimental condition. Therefore, the results partially support our hypothesis. There were also differences in the fluency and elaboration across different storytelling stages, in which more ideas and higher elaboration are present during the climax stage of the stories where peaks of action are prone to occur.

Creative Product

Given that CAT relies on a consensus regarding the perceived creativity of a given product, we started by calculating the inter-judge reliability of the ratings regarding the final stories created by children. Cronbach’s coefficient alpha showed a moderate agreement between judges, \( \alpha = 0.68 \) and \( \alpha = 0.70 \) for the experimental and control conditions respectively (McHugh, 2012). Results in terms of the perceived creativity showed no statistically significant difference between conditions, \( p > 0.05 \), with mean ranks of 47.02 and 43.98, for the experimental and control conditions, respectively. Therefore, this result does not support our hypothesis.

Discussion

In this work, we evaluated the impact of a new creativity support tool, CUBUS, on children’s creativity during a storytelling task. The effectiveness was evaluated in terms of the creative person, creative process, and creative product. Results concerning the creative process of children showed that children’s fluency, i.e., number of ideas generated during the process of creating a story, is higher in the control condition; results also showed that the originality of the ideas produced about the characters during the creative process was higher in the experimental condition. This result translates the paradigm of quality versus quantity since children generate fewer ideas, but the ones generated are more original. This result seems to be in line with the idea that when ideas are generated under creative tasks or contexts, they are deemed more unique, even if they appear in less quality (Derks and Hervas, 1988; Wierenga and Van Bruggen, 1998). Additionally, it makes sense that ideas about the characters in the experimental condition were more original since they were designed to provoke creativity in children through their emotionally expressive interactions. No other significant results were found.

Implications for Designing Tools for Creativity

Our study seems to support that when children play with tools or toys that convey and express emotion during storytelling, they engage in a more creative process. Specifically, this can inform toy designers to incorporate an emotional component to the new and interactive toys created for children. Additionally, this study shows that children can use minimalistic shapes, such as cubes, to create complex stories. This shows that simplicity in the design can provide engagement and interest in children towards a more abstract play in which the characters were non-stereotyped.

Conclusion

Our main contribution was the investigation of the role of emotionally expressive virtual agents in the social creativity of children during a playful activity. This activity revolved around a storytelling task in which children had to create a story using the emotionally expressive characters as their actors for the story. Due to the nature of the task, filled with creative potential by its open-ended and unrestricted creative process, children explored social creativity by engaging in collective creations of their stories.

Recommendations for future research

This study had several limitations that we would like to acknowledge. Regarding the lack of significant results concerning the effects of CUBUS in the creative person and the creative product, we attribute this to the small sample size in the evaluation study which consisted only of 20 children. Additionally, despite the study sessions being relatively long, children had a limited time to learn how to use the digital tool, which may have contributed to limiting the degree of creativity expression.

Although this study was conducted in school, it was performed in a private classroom which mainly replicates a lab
setting due to its controlled environment. We would like to evaluate CUBUS in informal school settings with children and evaluate its effect on their creative levels. Regarding the lack of results in terms of the creative person, we would also like to acknowledge that we have used a figural test TCT-DP, which evaluated the creative potential of drawing products to show creative levels. Since the main task that children engaged in was verbal as they created a story, this might not have been the more adequate test to apply in this context. A measure that can evaluate verbal creativity could have been more suitable taking into account the study design, as it can reveal important results, e.g., related to collaboration during a creative task (Kantosalo and Riihialo, 2019).

Additionally, in terms of the creative product, future studies should contemplate a larger sample size in terms of children and external judges. Another variable can also be the variation in expertise across the judge. We would like to note that datasets and supplementary materials of the study are released online in Open Science Framework5.

Acknowledgments

This study was funded by the Fundação para a Ciência e a Tecnologia (FCT) through grants numbers: SFRH/BD/110223/2015, UID/PSI/03125/2020, and UIDB/50021/2020. We express our gratitude to André Pires for the technical development of CUBUS.

References


5Supplementary materials can be accessed using the following weblink: https://osf.io/6cv93/?view_only=2c002947ad8046a0afa54b7a5ee4f5a9.
can be applied to most age and ability groups. Creative Child & Adult Quarterly.


Are machine learning corpora “fair dealing” under Canadian law?

Dan Brown  
Cheriton School of Computer Science, University of Waterloo  
Waterloo, Ontario, Canada  
dan.brown@uwaterloo.ca

Lauren Byl  
Library, University of Waterloo  
Waterloo, Ontario, Canada  
lrbyl@uwaterloo.ca

Maura R. Grossman  
Cheriton School of Computer Science, University of Waterloo  
Waterloo, Ontario, Canada  
maura.grossman@uwaterloo.ca

Abstract

We consider the use of large corpora for training computationally creative systems, particularly those that write new text based on the style of an existing author or genre. Under Canadian copyright law, a key concern for whether this is “fair dealing” is whether this usage will result in new creations that compete with those in the corpus. While recent law review articles in the United States suggest that training models on such corpora would be “fair use” in the United States, we argue that Canadian law may, in fact, forbid this use when the new products compete with works in the original corpus.

Introduction

The fair-dealing exception (section 29 – 29.2) in the Canadian Copyright Act (RSC 1985, c. C-42) allows the use of copyright-protected materials without permission or payment of royalties under certain circumstances. These circumstances include research, criticism, review, and private study, as long as what is done with the work is “fair.” For the purpose of criticism and review, proper citation of the copyrighted material is also required.

“Fair” dealing is not the clearest of concepts: There is no checklist with a target score that ensures that use is certain to be judged fair. Instead, there are a collection of factors that are evaluated as part of the review process. This collection of factors is not part of the Copyright Act. Instead, it was provided in a Supreme Court of Canada ruling, CCH Canadian Ltd. v. Law Society of Upper Canada (2004 SCC 13). The factors include the purpose of the use, the amount of the copyrighted material being used, and the effect of the use on the original work (for example, does the use compete with the market for the original work?).

Natural language corpora, as used by machine learning systems, form an interesting test case for this “fair-dealing” regime. Current state-of-the-art text-generation systems use truly massive data sets of human-authored text and generate text that is more and more like what a human would write. In 2021, such systems are trained on billions of words of human-authored text (Brown et al. 2020).

A rapidly growing paradigm in text generation these days is to use GPT-2 fine-tuned with an author-specific or domain-specific corpus. (See, for example, Lee and Hsiang (2019).) This approach yields texts with the surface features of an author/domain, while taking advantage of the high-grammatical fluency of transformers like GPT-2 (Radford et al. 2019). Therefore, this use can have two different components: modelling the corpus to inform the overall generation of the new text, and (possibly) inclusion of some short bits of the training corpus directly in generated works. (The current state-of-the-art system is actually GPT-3, which is fairly hard for most researchers and companies to access; in contrast, GPT-2 is readily available.) The resulting documents are often surprisingly hard to distinguish from human-authored text, although many still require a fair amount of human editing and correction. One example was an op-ed written by GPT-3, titled “A robot wrote this entire article are you scared yet, human?,” published in The Guardian in September 2020, but the true story of a lot of these products involves a huge amount of human massaging and shaping (Uitdenbogerd 2020; Jordanous 2017).

Many corpora for these systems are created by Web crawlers. This is certainly the case for the base corpus on which GPT-2 is trained, and could be true for other creative systems; for example, a poetry generator might be trained with user-submitted poetry on a poetry forum. This raises the question of the copyright status of the source documents, which may be set by the organizers of these fora themselves, in their own policies.

All of this leads to the question: Is creating a machine learning corpus “fair dealing” if the material contained in it is itself copyrighted? Here, we investigate this question by looking at the factors defining “fair dealing” in Canadian case law, after briefly contrasting the situation in Canada.
with the “fair use” model used in the United States. We focus our consideration on a factor concerned with creation of new works that compete with the copyrighted works being used for training, and explore a number of scenarios for this use. Ultimately, we argue that such use may, in fact, be unfair, and give a brief discussion of the consequences of this conclusion.

Existing Literature on the Question

The question of copyright and large-scale corpora is not novel; copyright issues for translation corpora, for example, have previously been discussed (Wilkinson 2006), and there is a short primer on how natural language corpora intersect with German copyright law (DFG Review Board n.d./2017). The World Intellectual Property Organization has studied whether or not such corpora should be permitted under international rules (2019), and some experts have highlighted the rulings in the United States in both the Hathi Trust1 and the Google Books2 cases as showing that the American “fair use” concept allows for the creation of corpora for research and other purposes.

In the Canadian context, Craig (2020) has explored whether computer-authored texts deserve copyright protection (her opinion is that they do not), and has expressed her concern that current law does not make clear how infringement concerns, if they were held to be valid, could be targeted at infringing systems that, in her view, lack autonomy as creators.

Our question is a bit different, as we focus on computationally creative authoring systems. These have not been the focus of any Canadian legal scholarship that we can identify. We focus on Canadian law, because it governs the computational creativity research of the first author.

Fair Dealing, Not Fair Use

We specifically are asking about machine language corpora in light of the Canadian concept of “fair dealing,” not the “fair-use” exemption available to users in the United States (Title 17 USC §107). U.S. law allows for researchers to make copies of copyrighted materials, and in both the Google Books and Hathi Trust cases, large-scale digital analysis of corpora (for example, to enable search) was seen as non-infringing, as long as users were not gaining access to chunks of the copyrighted materials in those corpora that were commercially relevant, such as whole pages of books, or definitions found in dictionaries. Some legal academics in the U.S. have argued for a much wider exemption, both for research in general, and for text and data mining in specific, claiming that the fair-use principles allow for it because of the positive benefits to society of scientific research (Carroll 2019). Sobel (2017) argues that Artificial Intelligence (AI) is in a potential crisis if learning from corpora and creating derived works is not held to be fair use, and argues that current U.S. law, which he deems to support this use, in fact gives U.S. researchers a competitive advantage.

Very recently, the U.S. Supreme Court has also ruled (in Google v. Oracle3) that Google’s use of some Java application programming interface code was fair use, although the ruling did not answer the question of whether the code itself was copyrightable. This ruling is not directly applicable to the case of corpora and fair use, but nonetheless suggests a willingness to allow for technology innovation as fair use, consistent with Sobel’s hopes.

More Detail: Factors for Fair Dealing

The full list of valid contexts for “fair dealing” are: research, private study, criticism, review, education, satire, parody, and news reporting. Creating a text generator is likely not education. If the generator is writing news articles, critiques or reviews, then the use of source materials about the events being reported on, or the work being reviewed or critiqued is fair dealing, but the base corpus being used to train language-model parameters is not itself being used in reporting, criticism or review.

Most of the time, a text generator is also not private study. Satire and parody may be the underlying goal for some text generators (e.g., mashups of the works of H.P. Lovecraft and the King James Bible, for example, Stross (2013)), but these are not the most common, and do not form the basis for the examples we discuss later in this paper.

The “Research” exception to the Canadian Copyright Act is broadly defined by case law; the ruling in CCH v. LSUC (2004) holds that “The fair dealing exception under s. 29 is open to those who can show that their dealings with a copyrighted work were for the purpose of research or private study. ‘Research’ must be given a large and liberal interpretation in order to ensure that users’ rights are not unduly constrained. I agree with the Court of Appeal that research


is not limited to non-commercial or private contexts.” (at ¶ 51). As such (and see below as well), “research” probably does include commercial research, although the extent of the researchers is clearly relevant.

Once the use is assigned to the general category of “research,” we must look at the various factors typically assessed by Canadian case law. The following purposes were laid out in the CCH v. LSUC ruling: “[T]he purpose of the dealing, the character of the dealing, the amount of the dealing, the nature of the work, available alternatives to the dealing and the effect of the dealing on the work are all factors that could help determine whether or not a dealing is fair. These factors may be more or less relevant to assessing the fairness of a dealing depending on the factual context of the allegedly infringing dealing. In some contexts, there may be factors other than those listed here that may help a court decide whether the dealing was fair.” (2004, at ¶ 60).

1) The purpose of the dealing: What is the user’s motive? This factor looks at the user’s “purpose or motive in using the . . . work” (¶ 54).

This is complicated. Most academic research is not directly commercial, but a lot of the use in the context of non-profit work is itself building data sets that will be used commercially. This may create situations where the motive changes over time. The boundary between non-profit and commercial research is fuzzy these days, and particularly in AI.

2) The character of the dealing: What was done with the work? Was it isolated or an ongoing use? How widely was the work distributed?

This is also complicated. The usage is often a single-use event (e.g. to fine-tune the language model), but the resultant parametrized model is repeatedly used, possibly with different prompts. The actual activity is usually to model sentence structure and how sentences flow from one sentence to another, but in practice, since the parametrization of a deep learning model is impossible to easily describe, it is entirely possible that chunks of the work will be directly “copied” into the parameters of the model, and may appear in the resultant generated texts. These occurrences of small chunks of copying would likely be an “insubstantial use,” and therefore, not require copyright permission, but it is unpredictable whether or not they would happen, or how often. The copyrighted work in the corpus typically would not be distributed, but the parameterization, in the form of the structure and weights of a neural network model, typically would be.

3) The amount of the dealing: How much of the work was used? How important was the content that was used?

Typically, the entire work is used; the goal is to have as large of a corpus of domain-specific writing as possible.

4) Alternatives to the dealing: Could a different work have been used?

This, again, is complicated. The goal of these generative models is to use as rich of a corpus as possible: Using such a corpus allows for more of the natural flow in an author’s writing to be modeled. So, if the goal is “as refined an approximation as possible,” then, no, a different work could not have been used.

5) The nature of the work being used: Does dissemination aid the public interest?

No. The work is, in fact, not being disseminated via this kind of modeling; to the extent that it is found within the parametrization, recipients of the model could not reconstruct the original work.

6) The effect of the dealing on the original work: Does the use compete with the market of the original work?

This is possibly the most complicated question. In the next section, we look at this question in more detail, exploring a variety of computational creativity scenarios.

Competitive Use and Computational Creativity

To the question of “does the use compete with the market of the original work?”, the answer is especially complicated. In the CCH v. LSUC (2004) ruling, it was made clear that this is not merely an abstract question: Copyright owners have to supply evidence of harm to their market because of the use in question. The onus of proving that dealing is fair is on the user, but such users typically lack the ability to see sales figures. “If there had been evidence that the publishers’ markets had been negatively affected by the Law Society’s custom photocopying service, it would have been in the publishers’ interest to tender it at trial.” (¶ 72). In a follow-up ruling, Alberta (Ed) v. CCLA (2012), the mere fact that sales have declined was not taken as sufficient proof that the use of copyrighted materials was a material factor in proving unfair use: “[O]ther than the bald fact of a decline in sales over 20 years, there is no evidence from Access Copyright demonstrating any link between photocopying short excerpts and the decline in textbook sales.” (¶ 35).

With this context in mind, we can consider a variety of ways works might be used in corpora, and examine whether these uses would compete in the market with the original:

1) Automated news-writing systems. The system that is built certainly competes with the journalists whose work is being used, but likely is not competing with the specific articles of the corpus, since the news-writing system will create stories about events occurring after the articles it was trained on. The text humans write about individual news events may be copyrighted, but the facts themselves are not protected by copyright. The system may compete with the owners of the copyright (the newspapers in which the work appeared), as well: If a start-up using a computational news writer builds a corpus of articles from The Globe and Mail,

that start-up’s product may compete with the *Globe*. Systems that write news editorials are closer to the creative writing systems discussed next.

2) **Automated creative writing systems.** The system that is built will create works in the same genre as the original work; in fact, they may be even designed to mimic a single author’s style. As of 2021, such systems are mostly curiosities: They require quite a bit of editing to make song lyrics (Ackerman and Loker 2016) or newspaper op-eds, and again, the training data are about moments in the past, but they are progressing quite speedily. It is entirely possible that in certain genres or domains, consumers will (knowingly or not) purchase computer-generated works, trained on corpora of human-generated work in the same field, in preference to new or existing work by humans.

To that end, yes, the systems that result from corpora can compete with the market of the original work. We note that this overall space is huge: systems that generate text, music, visual art, dance patterns, and more. This suggests that if the training data are copyrightable, the same overall questions as we are raising in this paper probably apply more broadly to other domains.

3) **Automated patent generation.** The goal of these systems is to create a patent “autocomplete” system which, trained on a corpus of patents, can start with the preamble of a patent and generate text that belongs in the patent (Lee and Hsiang 2019). This is obviously a remarkable goal, but here, again, the extent to which the new inventions might compete with the existing ones is likely to become more of an issue in the future. More generally, automatic legal authoring systems are definitely starting to come into their own, drafting motions, for example (Hudgins 2020), and their work absolutely competes with the work of human lawyers.

4) **Automated scientific paper generation.** Again, this is a blue-sky idea. We would be delighted if we did not have to chase down the citations this short article requires, and instead, a computer did it for us. But in theory, one could write an abstract and the paper could write itself around the abstract. Previously, this has mostly been used for satire, not for real research. Regardless, it is entirely possible that the existence of computationally generated papers would make the peer-review process collapse due to the required labour to assess all of those new papers, or that such papers might well compete with those in the corpus, particularly if the corpus was made of papers that “stand the test of time,” or the like.

In this manuscript, we do not consider chatbots. In this case, the system probably does not use copyrighted materials much, and will be trained using transcripts of natural dialogues, or successful customer-service interactions, or similar sources. Chat bots are also not consistent with the rest of the frame of this article, as the model will involve a lot more discourse analysis and (often) more of a model of what customer-service interaction a human participant is seeking to resolve.

## Conclusion

Overall, then, the question of competition between the result of a corpus’ use and the corpus itself is complicated, but particularly thorny in the context of creative-writing systems. Ultimately, if computational creative-writing systems become successful, then such systems will be competing with exactly the producers (and quite possibly even the creative artifacts themselves) through which the systems were trained. Rather than buying a greeting card for your spouse’s birthday, you might just send an automatically generated message, with a cheery computer-generated video based on a corpus of existing greeting cards. Rather than using existing pop songs as the soundtrack for a promotional video, you might use a “new” song, whose lyrics are produced by an engine trained on a corpus of existing pop songs. Rather than buying a copy of a Newbury winning book for your grandson’s birthday, you might buy a book written by a computer, using a corpus of existing children’s books. The possibilities are endless.

As such, producing corpora for training creative-writing systems will, over time, diminish the market for the copyrighted works in those corpora. Thus, building corpora of copyrighted materials for the purpose of training machine language models that compete in the same market as the training materials is unlikely to be fair dealing under Canadian law, particularly when it is practiced in commercial research.

This state of affairs places Canadian researchers at a disadvantage compared to researchers from other countries, most notably the U.S. If, as Sobel (2019) argues, U.S. law enables training of machine learning models from large collected corpora, and Canadian law does not, then researchers, such as the first author, must either gain permission to train models from copyright holders, use only materials with open licenses or in the public domain, or risk infringement lawsuits. We terminated a recent project, in part because gaining a good corpus of non-infringing materials was not easy. Either a change is needed in the Canadian copyright regime, or certain research may be chilled.

## Acknowledgements

The work of authors DB and MRG is supported by the Natural Sciences and Engineering Research Council of Canada.

---

5 For example, SCIgen ([http://pdos.csail.mit.edu/scigen](http://pdos.csail.mit.edu/scigen)), an automatic CS paper generator, was used to generate a submission that was accepted to a predatory CS conference in 2005 (Ball 2005).
References


Alberta (Education) v. Canadian Copyright Licensing Agency (Access Copyright), 2 SCR 345 (Supreme Court of Canada 2012). https://canlii.ca/t/fs0v5


CCH Canadian Ltd. v. Law Society of Upper Canada, 1 SCR 339 (Supreme Court of Canada 2004). https://canlii.ca/t/1glp0


Craig, C. 2021. AI and Copyright. in F. Martin-Bariteau and T. Scassa, eds., Artificial Intelligence and the Law in Canada (Toronto: LexisNexis Canada, 2021)


dazzling-results-but-some-assembly-is-still-required-146090


The stone of madness meets AI

Rafael Pérez y Pérez1, Lilla Lo Curto2, Bill Outcault2
1Departamento de Tecnologías de la Información
Universidad Autónoma Metropolitana, Cuajimalpa
Ciudad de México
rperez@cua.uam.mx
2Independent artists
New York, USA
locurto.outcault@gmail.com

Abstract
In this paper we argue that the computational creativity community would benefit by working together to create a methodology for a systematic assessment and comparison of our creative agents. To achieve this goal, we suggest collaborating with experts in real-world environments to develop what we refer to as Rich Creative Environment Experience (RICEE). We describe a work in progress, where two artists and one CC scientist collaborate, to illustrate a possible way to build a RICEE. Based on this experience, we suggest some initial steps to start the development of this methodology. Thus, the purpose of this paper is to motivate the CC community to collaborate in the development of a methodology for the design and implementation of RICEEs.

Introduction

Pérez y Pérez and Ackerman (2020) point out the importance of bringing computational creativity (CC) artifacts to the general public. They claim that this practice provides insights that are necessary for the advancement of the field. Similarly, we believe that it is important to take our autonomous creative systems to collaborate with experts (not related to CC) in the real world, away from the laboratories and, if you allow us the metaphor, away from the “comfort zones” where they are usually tested. In this way, it is possible to assess their capacities. We refer to the process of taking a creative agent to collaborate with experts in the real world, in an environment that is different to the environment where the program is normally used, as Rich Creative Environment Experience (RICEE). This approach has three main features: to be able to manage the complexity of real-world situations is a great challenge to any creative system; to collaborate with experts provides a different perspective of the role of creative agents and it is a priceless source of knowledge; to work away from the familiar “comfort zones” provides a context that allows testing the foundations of our systems. We claim that RICEE can provide insights for CC that are hard to obtain in a different way. However, in order to be useful for the field, RICEE requires a methodology that allows testing and comparing different programs in a systematic way. Unfortunately, we are far from reaching that goal. The purpose of this paper is to motivate the CC community to collaborate in the development of a methodology to design and implement RICEEs. With this aim in mind, we share the work that we are currently developing and that was the origin of all these concerns.

Our current project explores how to represent in computer terms part of the knowledge employed by artists during the production of a piece in order to inform the generation process of a computer creative agent. Let us elaborate this idea. Artists employ sketches, storyboards, drafts of texts, and so on, to plan the development of a creative artifact. This material characterizes an important part of the artists’ perspectives, expectations and beliefs about the piece in progress. We consider that, if we can represent in computer terms part of that experience, and we feed a creative agent with this information, then, the system will be able to generate outputs that the human artists will find interesting enough to be included as part of the piece in progress. Thus, rather than trying to build a system that collaborates in real time with humans, we are interested in developing an autonomous creative agent that works independently, but whose generative process is influenced by a representation of part of the knowledge that artists employ for realizing a particular creation (although this paper is about creative systems that were originally designed to work alone, we believe that the same or similar ideas can be applied to collaborative systems).

For this project, two multimedia artists and one scientist in computation creativity have gathered together. We employ MEXICA, a computer system that generates narratives, as the creative agent; and Stone of Madness, a
multimedia installation, as the artistic work in process. The project is divided in the following steps:
1. The artists write drafts of the texts to be used in Stone of Madness.
2. These texts are adapted into a description that MEXICA can process.
3. MEXICA uses these descriptions to build its knowledge-base.
4. As a result, MEXICA generates new narratives that are shaped by the artists’ representation of knowledge for Stone of Madness.
5. Some of MEXICA’s narratives are selected by the artists to become part of Stone of Madness.

Step 1 is already finished. We are currently working in steps 2 and 3. We still need to develop and test steps 4 and 5. This paper describes the core features of Stone of Madness and MEXICA, describes the challenges we are facing at this time and finally reflects about the contributions of this work to develop a methodology for RICEE.

Stone of Madness

The title of the in-process art installation, Stone of Madness, is based on the medieval belief that a stone was inside the head of mentally ill people, causing their behaviors, and it required extraction through trepanation. The project is based on the artists’ past work concerning the fragile nature of life and some personal history stemming from one of their pasts. Her parents had immigrated to the United States from Sicily and her father, a psychiatrist, worked with the criminally insane in state hospitals. She spent her childhood living on the grounds of state asylums, primarily Arizona State Hospital, located in Phoenix, Arizona, at a time when psychiatric medications were only beginning to be employed and when frontal lobotomies were still considered a viable treatment. There she mixed fairly freely with the patients and was exposed to quite varied perceptions of reality. Based on this personal history and the artists’ focus on physical and psychological vulnerability they thought it would be an interesting project to use the narrative of this history, both in text and verbal forms, together with animations, to develop a multi-media installation. Their intention was for the work to explore some of the conflicts and connections between the distress and stigma of mental illness, the perception of it by its sufferers and those around them and the mental institutions themselves, all from the point of view of a little girl. As their thoughts developed about the installation, they began to feel a linear narrative could be too rigid for the project. In researching various systems to produce human-like text, however, they found those systems were not able to create the coherent sequences of narration they felt the project required. Because the process of artistic creation involves multiple dimensions, data alone cannot develop systems that explore in depth the relation between art and AI. Also, because of the psychological nature of the project, they wanted the narrative to be more ingenuous, as if the story was being told by a child. They were looking for a method where the bones of the story could be used as the basis for a system to develop its own novel narratives.

MEXICA

MEXICA is a computer model of the process of creative writing (Pérez y Pérez and Sharples 2001). The system represents knowledge in terms of emotional relations and conflicts between characters. This knowledge is registered in structures known as Contextual Structures. This is an example:

Contextual structure 1.
When the health of character A is at risk [this is a conflict], the narrative might continue as follows:
Character B arrives and cures Character A
Character A dies
Character B steals Character A’s possessions

Character A and Character B are variables that can be substituted by any character in the tale. The first part of the Contextual Structure represents a specific situation in the story in terms of emotional relations and conflicts between actors, and the second part represents possible ways to progress the tale given that situation.

MEXICA builds its knowledge base from two text files provided by the user of the system, known as the Dictionary of Story-Actions and the Previous Stories. The Dictionary includes all the actions that characters can perform; the Previous Stories include a set of narratives, written by humans, following a rigid format, that the system employs to build the Contextual Structures. The number and features of the Contextual Structures depend on the content and length of the Previous Stories (for details see Pérez y Pérez 2007). When MEXICA is developing a new narrative, it looks for Contextual Structures that are equal or similar to the current story context to decide how to progress the tale. In this way, the Previous Stories strongly influence the kind of narratives that MEXICA generates (for an analysis of the influence of the Previous Stories in the outputs generated by the system see Pérez y Pérez 2015 and Guerrero and Pérez y Pérez 2020).

Extracting the Stone of Madness from MEXICA

The goal of this project is to employ some of the texts written for the piece Stone of Madness to build the set of Previous Stories that MEXICA employs. Then, we will ask the system to generate new narratives that, hopefully, will be interesting enough to become part of the final version of
the piece. In the following, we describe some of the challenges that MEXICA faces to contribute to the piece Stone of Madness.

Type of story-worlds. The narratives in Stone of Madness mainly take place in a psychiatric hospital in the USA; the characters are doctors and their families, nurses, patients, and so on. The narratives generated by MEXICA take place in a pre-Hispanic environment; the characters include tlatoanies, jaguar knights, princesses, and so on. Thus, it is necessary to modify the system to manage this new type of story-worlds.

Types of narratives. MEXICA generates plots where an actor faces an obstacle that has to be overcome, e.g., the jaguar knight has to rescue the princess. Thus, the narratives in MEXICA have an introduction, development of the conflict, a climax and a resolution. By contrast, the narratives in Stone of Madness describe ordinary situations in a mental hospital, told from the perspective of a little girl, that make the reader question the behavior and attitudes that doctors and nurses have towards the patients. These differences in the type of narratives present an important challenge for us. Because in MEXICA dramatic situations (e.g., wounded or kidnapped characters) are used to guide the unraveling of the tale, the features of the narratives in Stone of Madness require reconsideration in the generation process in MEXICA.

Representing the Previous Stories. The process of adapting the texts of Stone of Madness into descriptions that MEXICA can process has several challenges. This is an example of one of the texts written by the artists:

“My parents immigrated to the United States from Sicily with my brother, making a stop in Caracas where I was born to wait for entry into the USA. My mother was a housewife and my father was a physician specializing in psychiatry. After arriving in the US he began working in state hospitals for the mentally ill and the criminally insane. My family lived, and I grew up, on the grounds of various asylums, mingling with the patients and experiencing all kinds of different realities. We arrived in NY where my father found work at Pilgrim Hospital in Long Island but he wasn't fond of the east coast and so we headed west and eventually settled at Arizona State Hospital in Phoenix, AZ. We were given a small brick house on the hospital grounds where all the doctors lived. There was a long chain link fence that separated the doctor's families from the patients living in the hospital and the children were allowed to play within this area. My friends were not only other doctors' children, but also some of the patients who mingled around the fence. These patients weren't dangerous but some of them did things to themselves that most people would consider in bad taste. I would watch and learn from the patients and they became friends. Unfortunately, my parents became aware of

my relationship with the patients and, since they found the patient's actions inappropriate, told me that I shouldn't mingle with them. Since I considered them my friends, this was hard to do, especially since I was having a hard time distinguishing how they differed from the supposed “normal” on my side of the fence. Needless to say, I continued to see my friends....”

Because MEXICA does not work with natural language processing, the system cannot handle this type of narrative. Thus, it is necessary to transform this text into a description that the system can manage.

The artists chose a small part of the original narrative (written in italics) and rewrote it as a sequence of events; each event includes only one action (which was included in the Dictionary of Story-Actions), some characters (mother, father, Lilla, patient) and, optionally, the name of the family. The following shows the result of this process (story-actions are represented by words separated by a hyphen):

Story 1
Scenery: Family House
Father and mother Were-the-parents of the Family LoCurto.
Lilla Was-the-daughter-of mother and both were members of the Family LoCurto.
Mother felt-uncomfortable with the patient.
Mother and Lilla walked-together-to-the-fence.
The patient also went-to-the-fence.
Lilla mingled-with the patient.
As a result, mother strongly-disliked the patient.
Lilla and the patient became-friends.
Mother felt-upset-with Lilla.

MEXICA is able to read this description and to build its contextual-structures. We repeated the same process for the whole original text written by the artists. We ended up having several Previous Stores. We need to test if the knowledge-structures that the system builds from these stories are enough to produce narratives that satisfy the artists’ requirements.

Knowledge structures. The narratives in Stone of Madness require representing emotional relations and conflicts between characters that are not contemplated in MEXICA. For instance, to make sense of the narrative just described, one needs, at least, to have a basic understanding of the concept of family as a social group, to know who the members of a family are and what their roles are. In this way, one can figure out that, because the mother feels that the patient is a threat to her daughter, she attempts to stop Lilla from mingling with the patients. Similarly, it is necessary to comprehend that in this social group, the mother has a higher hierarchy than the daughter. That is why she tries to impose a specific conduct on the girl.
In order to be able to employ this narrative as one of the Previous Stories, MEXICA needs to represent social groups like families, and to characterize emotional relations and conflicts that might emerge between members of the family. Based on an analysis of the preceding story, it is possible to determine some family conflicts. For instance, when one member of the family is fond of an outsider (a character that does not belong to the family) but other members of the same family with higher rank feel threatened by this outsider, a tension between the members of the family arises. The challenge here is to establish how these new knowledge-structures will be created and how they will be used during the generation of a new tale.

**Discussion**

This paper describes a project in progress where an automatic narrative generator contributes with some texts to the piece Stone of Madness. Our aim is to employ this experience to show an example of a RICEE. Hopefully, this example provides insights about how to develop a methodology for the design and implementation of RICEE. For this project, we have performed three basic steps. We reflect on them as a first phase to build the methodology: (1) To establish the limitations of the creative system. Defining at least three categories that describe the limitations of the system would be useful for a methodology. For instance, limitations associated to the outputs, limitations associated to the knowledge-structures, and limitations associated to the generation process. In this way, it is possible to have a clear picture of the scope of the creative agent.

(2) Description of the RICEE. A methodology requires a description of the artistic project chosen for the design of the RICEE, and details about how the creative agent will collaborate in the piece. The purpose is to have a clear understanding of what the human artists expect from the agent.

(3) Defining the RICEE’s challenges. As a result of contrasting the outcomes of steps (1) and (2), it is possible to identify the computational and creative challenges that the RICEE provides to the system. These challenges might be organized based on how complex their implementation is, or the kind of new abilities that they provide to the system.

The piece Stone of Madness was planned without having MEXICA in mind. Based on the three steps mentioned earlier, we conclude that the main targets of this project are:

- How to create a new story-world.
- How to develop a method to transform real story-world narrations into descriptions that MEXICA can handle.
- How to incorporate social structures, e.g., family groups, into the system.

- How to develop new knowledge structures that represent emotional relations and conflicts between the members of a family.
- How to handle new narrative structures during the creation of the knowledge-base.
- How to drive the generation process for novel narrative structures.

All these are new situations never contemplated in the original design of the system. We believe that, when this project is finished, the results will be useful to test the theories about creative writing behind the design of the MEXICA system. In this way, we will be able to create more elaborated narrative generators.

We are convinced that collaboration with artists and other experts can be useful for the advancement of CC if we are able to develop a methodology that allows for the systematic design and implementation of this type of environment. If members of the CC community share their own experiences working in projects similar to the one we have described here, we believe that we will be able to create a common mechanism to test and compare our systems. The goal of this paper is to motivate the CC community to collaborate in the creation of a methodology for the development of Rich Creative Environment Experience (RICEE). The reviewers of this paper have taken the first steps in that direction. They have pointed out the necessity of considering the following aspects:

- RICEE involves an assessment from experts. It is worth reviewing the work on expert and non-expert evaluation (e.g., Kaufman et al. 2008; Lamb et al. 2015)
- It is necessary to assess if it is possible to compare systems’ limitations between diverse CC disciplines.
- RICEE will benefit from ethnographic interviews by a third party about the experience, treating the whole affair as a researcher-system-artist collaboration.
- "One might call close cooperation with artists "ethology of the creative process," where computer scientists confronted with an ongoing process of creation of human agents will be better positioned to develop new and fresh ideas on how to improve autonomous systems" (anonymous ICCC’21 reviewer).

Thanks to the reviewers for these useful contributions. Hopefully, soon you will let us know yours.

**References**


The 7th AISB Symposium on Computational Creativity (CC@AISB2020/21)

Anna Jordanous
School of Computing
University of Kent, UK
a.k.jordanous@kent.ac.uk

Juan Alvarado
School of Electronic Engineering and Computer Science
Queen Mary University of London, UK
j.alvaradolopez@qmul.ac.uk

Abstract
The 7th AISB Symposium on Computational Creativity (CC@AISB2020/21) took place online during April 2021. As event organisers, we report on the event, its ongoing history at the AISB and its future.

Introduction
This year’s Computational Creativity Symposium at AISB is the seventh such symposium to take place as part of the annual convention of the Society for the Study of Artificial Intelligence and Simulation of Behaviour (AISB 2021) which was held online from 7 – 9 April 2021.

This annual symposium typically features a number of presentations over a single day, covering a range of topics in the field of Computational Creativity. Issues addressed include practical work in the area, theoretical approaches to creativity, and philosophical questions raised on the potential of non-human creative agents.

Held online, the 7th Computational Creativity symposium at AISB (CC@AISB’2020/21) was originally scheduled for an in-person event in London in April 2020. Due to COVID-19, the event was postponed one year and held online on 9th April 2021, as part of the AISB Convention.¹

Over the last few decades, computational creativity has attracted an increasing number of researchers from arts and science backgrounds, from academia and industry. Philosophers, cognitive psychologists, computer scientists and artists have all contributed to and enriched this field.

This symposium aims at bringing together researchers to discuss recent technical and philosophical developments in the field, and the impact of this research on the future of our relationship with computers and the way we perceive them: at the individual level where we interact with the machines, the social level where we interact with each other via computers, or even with machines interacting with each other.

This year we were delighted to also run a Show-and-Tell demo session as well as the paper presentations, showcasing demonstrations of computational creativity research results as well as more traditional talk presentations.

¹Hence the acronym CC@AISB’2020 refers to the event due to take place in 2020, and the acronym CC@AISB’2020/2021 the rescheduled event that took place April 2021.

This year’s event, CC@AISB2020/21, has a web presence at https://sites.google.com/view/aisb2020cc/.

A quick history of computational creativity research at AISB
AISB is the Society for the Study of Artificial Intelligence and Simulation of Behaviour. Founded in 1964, it is the world’s oldest AI society, the UK’s biggest AI society, and a member of European Coordinating Committee for Artificial Intelligence (ECCAI).² Based in the UK with international membership, AISB’s annual convention is the longest running convention on Artificial Intelligence. The annual AISB convention consists of various symposia, organised as standalone events within the convention. This year saw the 7th running of the Computational Creativity symposium.

AISB’s historical connections to computational creativity research include having Margaret Boden (Boden 1992) as one of its Fellows for many years. Connections to computational creativity research have also arisen recently through AISB’s organising of the Loebner Prize (a formalised Turing Test competition). In 2019, after the passing of Loebner Prize sponsor Hugh Loebner, AISB continued the Loebner prize event by incorporating it as a part of an event AISB X: Creativity Meets Economy with the theme and an “exhibition of art made by and with computers”.

AISB occupies an important part of the history of computational creativity research, hosting computational creativity research events multiple times in the past 25 years. Creativity was the theme of the entire AISB’99 Convention, which took place in Edinburgh, UK, organised by Geraint Wiggins, Helen Pain and Andrew Patrizio. Since then, Computational Creativity workshops have been held in conjunction with several AISB Conventions:

2000 Creative & Cultural Aspects of AI & Cognitive Science
2001 Symposium on AI and Creativity in Arts and Science
2002 AI and Creativity in Arts and Science
2003 AI and Creativity in Arts and Science

Largely, the team driving these AISB events (and computational creativity workshops at other conferences) were becoming more involved in working towards an integrated in-

²https://eccai.org
³(https://aisb.org.uk/aisb-events/)
The international research event on computational creativity, leading to the establishment of the annual International Conference on Computational Creativity series in 2010.

Soon a desire grew for additional events in computational creativity to complement ICCC. The current series of Computational Creativity symposia at the AISB Convention started in 2014, and since then, there has been a Computational Creativity symposium at every annual AISB convention. This year’s symposium was the 7th such symposium.

The symposium respects AISB’s traditional positioning of the convention as a venue that encourages reports of work in progress alongside presentations of more mature research. As the AISB Convention Handbook (AISB Committee 2015) recommends:

“the general principle should be that relevant submissions that represent a decent degree of novelty, are not published elsewhere, are well presented and reasonably rigorous should be accepted.

The AISB wishes accepted submissions to be at a high international standard of quality. However, symposia will often be workshops that air work in progress or exist primarily to stimulate discussion, set new agendas, forge new interdisciplinary links, etc., and an appropriate level of flexibility can be allowed in reviewing.”

Full papers are usually required, though this year we also encouraged shorter submissions of ‘Show-and-Tell’ demo papers, inspired by ICCCs in previous years. Proceedings of each year’s workshops are available via https://aisb.org.uk/convention-proceedings/ and https://aisb.org.uk/past-conventions/, and typically receive good citation rates within computational creativity research.

Typically, the symposium hosts a keynote speaker each year. In past years, many key figures in computational creativity research have given CC@AISB keynotes:

CC2014 - keynote given by Geraint Wiggins
CC2015 - keynote given by Tony Veale
CC2016 - keynote given by Margaret Boden
CC2017 - keynote given by Simon Colton
CC2018 - keynote given by Pablo Gervas
CC2019 - keynote given by Jeremy Gow

This year’s symposium was organised by Juan Alvarado and Anna Jordanous. Past organisers of the Computational Creativity symposia¹ have included Mohammad Majid al-Rifaie, Jeremy Gow and Stephen McGregor. Al-Rifaie and McGregor have also been responsible for organising and editing three special issues of Connection Science journal in 2019 (vol 31 (1), (Al-Rifaie 2019)), 2017 (vol 29 (4), (al-Rifaie and McGregor 2017)) and 2016 (vol 28 (2), (al-Rifaie 2016)), featuring developments of computational creativity work published at CC@AISB. Computational Creativity research was also featured as part of the special issue of Cognitive Computation vol 4. (2012) (Bishop and Erden 2012) on “Computational Creativity, Intelligence and Autonomy”, arising from an AISB symposium on Philosophy and edited by J. Mark Bishop and Yasemin Erden.

The 2020 symposium (postponed to 2021)

Symposia for AISB are proposed each year to the central convention organisers, and for the 7th year running, a successful Computational Creativity symposium proposal was made. With a program committee of 13 excellent reviewers, we accepted five full paper submissions and three Show and Tell extended abstract submissions for CC@AISB’2020.

Due to take place in London in April 2020, like many 2020 events the AISB Convention was cancelled due to the rising situation with COVID-19. As this was at the start of the pandemic in the UK, it was one of the first events to fall during lockdown. First we lost our keynote speaker François Pachet, who quite rightly felt uncomfortable travelling to the UK. At this point, we still hoped the convention would go ahead, and were grateful to Colin Johnson for accepting our short-notice invitation to give a keynote instead. With participants coming from across the world, though, slowly more participants decided not to travel to London. By April 2020, the UK was in lockdown and AISB’2020 was put on hold.

At this point in time, still early in the pandemic, there was not the knowledge that we have now about running online events. Hence AISB Convention was postponed. With no knowledge of how long the pandemic would be affecting our lives, we believe that at this point many people hoped the AISB Convention (and CC@AISB’2020) could go ahead as planned in London later in the year. Instead, as we know, online conferences became the norm. The AISB took the decision to delay the entire Convention until April 2021. Even then, given the uncertainty whether we could have in-person events, or even hybrid events, it was only with the imposition of the third UK lockdown that the decision was taken to hold AISB’2020 (by this point, AISB’2020/21) fully online. We were delighted that all our original authors were able to attend, though very sadly due to time constraints in organisation and in the day’s scheduling, we were not able to re-invite our keynote speaker François Pachet.⁵

The 2021 (postponed from 2020) symposium

There was much discussion about when the postponed AISB 2020 Convention would take place. Eventually COVID-19 made the decision for convention organisers; the convention was to run fully online in April 2021. With this online format, registration fees were dropped, with all symposia made freely available to AISB members.⁶ At one stage, Alvarado and Jordanous were not sure if it would be possible to run the event in 2021; as well as both trying to keep up with the additional volume of work through online university teaching, Alvarado was in the final stages of writing up his PhD thesis and Jordanous had a large admin role for her department, completing the month before the revised dates for AISB 2020/2021. However, with support from the convention organisers and enthusiastic responses from symposium partic-

¹We are maintaining a links page to past symposia at https://sites.google.com/view/aisb2020cc/history-of-ccaisb.

⁵We do hope François Pachet can forgive our oversight in the rush of re-organising the 2021 event, and would be delighted to host him in a future year!

⁶Non-AISB members needed to purchase a year’s AISB membership to attend, however the monetary cost of AISB membership was considerably cheaper than the normal cost of registration.
ipants, the CC@AISB’2020/2021 convention went ahead on 9th April 2021, the final day of the convention.

Academic communities have learned a lot in the past year or two about the fatigue involved in participating in online events. Hence the AISB convention was scheduled as three days of sessions 13:00-17:00 UK time, with AISB plenary keynotes scheduled each day at 17:00-18:00. Each hour, we scheduled in a ten minute ‘comfort break’, to rest the eyes and walk away from our screens before returning for more talks. Each speaker had a slot of 25 minutes.

• First Experiments in the Automatic Generation of Pseudo-Profound Pseudo-Bullshit Image Titles (paper) (Colton, Ferrer, and Berns 2021). Colton started the CC2020/21 symposium with an entertaining, provocative talk including the point that ‘one person’s bullshit is another person’s high philosophy’. He reported how International Art English can be usefully employed to generate image titles in the spirit of ‘pseudo-profound bullshit statements’. This work contributes towards an app development project for casual creators to enjoy creating abstract images.

• Role-Based Perceptions of Computer Participants in Human-Computer Co-Creativity (paper) (Kantosalo and Jordanous 2021). Kantosalo presented a new categorisation of the roles that computers can take in human-computer co-creativity. The work aims to move us away from pre-defined (limiting) roles for a computer from creativity support tool literature to an increase in the amount of creative responsibilities taken on by co-creative computational systems. The endgoal is a wider recognition of what computers can contribute in terms of creative collaborations, taking fluid emergent roles. Some fascinating questions after the presentation probed about if we reach the stage where computers become the creative lead and humans essentially become the creativity support tool (a discussion which spilled onto Twitter7, and the implications for computer-computer co-creativity in terms of how we might get computational collaborators to flexibly change roles through interaction.

• Darwinian Creativity as a Model for Computational Creativity (paper)(Helliwell 2021). Helliwell presented her work on Darwinian Creativity (Simonton) as a model for computational creativity, asking whether this model could be applied as a tool for evaluation of computational creativity. Helliwell related the Darwinian model of creativity to GANs and CANs (Creative Adversarial Networks), then considered whether Simonton’s model could be applied as assessable standards for creativity evaluation.

• Will the real artist stand up? Computational creativity as mirror to the human soul (paper) (Parthemore 2021). In this philosophical talk, Parthemore offered and dissected a working prescriptive definition of creativity: “The at least partly - yet never fully! - intentional act of an intentional agent or agents recombining elements of past or present experience in more or less strikingly novel ways to yield insights - from the subtle to the life- or world-altering - or more immediate practical benefit.” Parthemore positioned Alan Turing as one of the pioneers of computational creativity, emphasising the links between creativity and intelligence, which sparked some discussion on Twitter 8. A particularly lively discussion followed this talk!

• Assessing Creativity of MEXICA: An Application of Ritchie’s Criteria (paper) (Alvarado and Wiggins 2021). Alvarado discussed using Ritchie’s criteria (Ritchie 2007) to revisit and assess the creativity of Pérez y Pérez’s MEXICA (Pérez y Pérez and Sharples 2001). Ritchie’s criteria have long been controversial as a measure of computational creativity systems. Alvarado demonstrated how they give us some valuable insights about typicality and value of system outputs; this paper uses them to allow MEXICA to test itself, to explore its conceptual space to optimise its performance.

• Walk the Line: Digital Storytelling as Embodied Spatial Performance (Show-and-Tell demo) (Wicke and Veale 2021). Wicke presented the first of our Show and Tell demos on embodied digital storytelling - sadly with robots sending their apologies due to COVID-19. Instead Wicke talked us through some videos on how spatial schemas can be used in gestures to enhance cross-modal communication during story-telling.

• AMI – Creating Musical Compositions with a Coherent Long-term Structure (Show-and-Tell demo) (Ma, Brown, and Vecchiotti 2021). Moving from stories to music in our Show and Tell demos - Ning Ma presented the AMI system for creating music using AI at CC2020/21. Ma presented AMI – Artificial Music Intelligence, a deep neural network that can generate musical compositions of different instruments with a coherent long-term structure. AMI uses a general-purpose deep neural network architecture. It predicts a music note with an autoregressive model depending on the last note and a long sequence of notes. They enhance the learning of musical structures by adding different kinds of embeddings: one short-term embedding and one long-term embedding. As a result, the model can maintain a coherent long-term structure and occasionally pick up different movements9.

• Jazzy Beach Critters: a Demonstration of Real-Time Music Generation with Application to Games (Show-and-Tell demo) (Quick and Burrows 2021). Our final CC2020/21 demo talk came from Donya Quick’s work on on-the-fly game generation - Jazzy Beach Critters10. They provided a proof-of-concept implementation to create an interactive scene where the music changes as the user interacts with creatures in the environment. This is a proposal for the generation of music in video games using procedural methods. They use improvisation models in real-time intending to generate variations in the music in response to the events triggered by the user in a way that the transition between music and moods of characters in the scene happens without abrupt changes.

7https://twitter.com/annajordanous/status/1380517501998600198
8Audio examples at: https://meddis.dcs.shef.ac.uk/melody/samples
9Online demo: http://donyaquick.com/jazzy-beach-critters/
Overall, the day’s talks and demos gave us a wide-ranging and thoroughly enjoyable tour of computational creativity technical experiments, demos, theories, frameworks and philosophy research findings, across domains ranging from music to robotic storytelling. It was a fascinating, thought-provoking day, with much to explore and learn from.

After the computational creativity symposium talks, many of us headed to the main AISB online channel for an excellent plenary talk for that day, given by Dr Sabine Hauert from University of Bristol, titled “Swarms for people”.

Moving forwards
Proceedings are published on the AISB 2021 website (Alvarado and Jordanous 2021). Videos of the talks are also available to AISB members, through the Members Area of AISB’s website. We are considering the publication of a selection of extended and re-reviewed papers from the symposium in a journal special issue.

We hope that the 8th AISB Symposium on Computational Creativity will run in 2022. AISB Convention organisers typically issue calls for symposium proposals to be submitted by around September of the year before the convention runs, and both current organisers are keen to see that a proposal is submitted for the Computational Creativity symposium. Alvarado and Jordanous welcome offers of assistance in proposing and (if successful) running the symposium in future years, or joining our excellent program committee.

We hope to continue the Show-And-Tell demos sessions, which we feel enabled new types of submissions and which opened the symposium up to new authors who might not have submitted to CC@AISB in the past.

Two areas we hope to improve in the future are the diversity in keynote speakers and financial support for the symposium through sponsorship.

- Diversity in keynote speakers: While we dearly hope that we can host our intended keynote speaker, François Pachet, in the near future, we also note that the majority of past keynote speakers have been white male speakers (all except Margaret Boden in 2016). Computational creativity is growing in diversity and we are keen to encourage this and make sure our range of keynote speakers better represents - and inspires - such diversity. We also hope to involve more industry speakers (such as Pachet), leading us onto the next improvement we wish to focus on:

- Sponsorship: An area that we would also like to improve is links with industry/non-academic organisation partners. The symposium receives limited financial support from the Convention, typically one free registration per symposium (and of course all in-person hosting costs and logistical arrangements covered!) Hence we are looking to secure sponsorship to cover expenses of inviting keynote speakers, to offer prizes for best papers, and in an ideal world, to offer some financial support to participants who might not otherwise be able to attend.

Looking towards the future of CC@AISB: at the time of writing, we do not know whether future AISB conventions will be held online, in-person, or with some hybrid arrangements. What we do know, though, is that computational creativity and AISB have a rich set of historical connections that we are proud to have played our part in continuing to 2021. Long may it continue!

References
5. Theory & Philosophy
Incorporating Algorithmic Information Theory into Fundamental Concepts of Computational Creativity

Tiasa Mondol
tiasa.ap.10@gmail.com

Daniel G. Brown
David R. Cheriton School of Computer Science
University of Waterloo
dan.brown@uwaterloo.ca

Abstract
Can we attribute creativity to an artifact by examining its computational history? What can be said about its value and novelty if the artifact is computationally laborious and interesting? Can the computation which gave rise to the artifact, help us interpret its artist? We look at these questions through the lens of Algorithmic Information Theory. Expanding on some of the advanced topics in this field, Kolmogorov Complexity and its conditional and resource-bounded versions, Logical Depth and Sophistication, we show that many of the standard criteria used in computational creativity follow naturally from these concepts. We also address the question of whether an artifact is a typical or novel creation of its artist, by first generating a good description of the artist’s known works and then examining how fit this explanation is for the new artifact. Although using Kolmogorov Complexity is not without its challenges, its incomputability being the obvious one, we show that by rooting our analysis into the algorithmic complexity of an artifact we inevitably shed light on some of the fundamental concepts of creativity: typicality, novelty, and value of an artifact, the creative process, and the creator.

Introduction
A key question since the rise of computational creativity has been which criteria characterize an artifact as creative, and how an algorithm can evaluate these criteria (Lamb, Brown, and Clarke 2018). In this work, we use the lens of algorithmic information theory (Li and Vitányi 2019) to look for universal properties that indicate whether a work is of high quality or value, novel or typical within a domain or a creator’s oeuvre. These commonly used criteria for creativity follow smoothly from several advanced concepts in algorithmic information theory. Quality is assessed by showing that there is a high amount of computational effort required by any short program whose output is a given artifact, while typicality and novelty are shown by looking for a program that can generate all members of a class of objects and assessing how random a new object is within that class. Another outcome of our work is that we can model the producer of an artifact by distilling down its core properties. Then based on the complexity and volume of these properties we can identify whether the artifact is a magnum opus of its artist.

Key to our work is a firm computational underpinning to some of the canonical ideas of creativity and to build it we carry out a more thorough exploration of Kolmogorov complexity and its adjunct ideas. In previous work, numerous authors have attempted to use two of the most basic ideas, the raw Kolmogorov complexity $K(x)$, and the conditional Kolmogorov complexity $K(x|y)$ of object $x$ given $y$ as a free input, as measures of the creative value of an object or of the similarities between two objects (Li and Sleep 2004; Ens and Pasquier 2018). While these measures are certainly important, their use in this manner is still potentially of concern: an object with low Kolmogorov complexity might or might not be trivial, and objects whose Kolmogorov complexity are close to their length (which is the maximum possible) are typically just random noise. Even looking for an object of “medium” complexity is insufficient: such an object might either be the product of substantial computational effort (which, in our telling, makes it a high-quality work) or could be trivial repeated patterns augmented with random bits to make it seem serious. Only by looking at the actual computational effort required to produce the object can we assess its quality; the raw value of $K(x)$ is insufficient. Similarly, while the conditional complexity $K(x|y)$ is important, more useful is to look at $K(x|M)$ where $M$ is a model representing the non-randomness in $x$. Then a good model for $x$ will be most successful in compressing $x$ and classifying it into a category that is most appropriate.

Here, we begin the build out of this theory of creativity of computationally created objects and how to assess them. We begin with an introduction to algorithmic information theory, where we focus on the concept of models that gives a two-part description for members of a genre or a creator’s oeuvre and how typicality or novelty can be assessed for an artifact. The concept of logical depth allows us to assess the computational effort required to produce an object: if an object with high compressibility has slow-running short programs, the most likely explanation for the object is that it required substantial effort on the creator’s part to produce it, and it is valuable. Additionally, the concept of sophistication allows us to identify the inherent challenge of representing the producer of an object: if the only good models for an object are of high sophistication then it must have been created by a skilled creator. These concepts have not previously been extended to the domain of computational creativity. At the end of this paper, we describe the relationship between these concepts to existing efforts in joining algorithmic in-
formation theory with computational creativity and aesthetics; we also give, in Table 1, an algorithmic recipe for our own approach which follows naturally from the theoretical concepts presented in this paper. Although it is necessary to navigate the subsequent sections carefully to fully understand the ideas in this table and our critique of existing approaches, interested readers are encouraged to consult it for a helpful summary of the topics we present hereafter.

**Kolmogorov Complexity and The Artifact**

We begin by building a formal description method for a creative product by associating it with a Turing-computable function. Although it is difficult to regard an artifact like “The Mona Lisa” to be an output of something as inanimate as a function, such formulation provides us with a powerful theoretical framework to analyse an object computationally. We work with Turing machines\(^1\) that compute partial recursive functions, which are only defined for some inputs and for which an effective algorithm with step-by-step instructions exists to compute it. The Turing machine upon receiving an input, computes the function on that input by manipulating the bits present on its tapes and halts with output \(x\) or runs forever if the input is undefined. Thus a finite binary program which encodes the Turing machine along with the input on which it halts with output \(x\) completes a description of the target creative product \(x\). It is important to note here that when talking about generating a creative product on a Turing machine’s tape, we are essentially reducing the vibrant object to a binary representation which may be a lossy depiction of the original product. Some creative objects are also a sum of their spatial and temporal contexts and cannot be perceived in isolation. For these cases, we assume that the target output \(x\) is a recognizable version of the original product which can be recovered from \(x\) within reasonable time bounds: for example, to generate a 2-dimensional \(m \times n\) painting, we can imagine a Turing machine outputting the RGB values of its each pixel. More important for us is to maintain an objective notion of computability and description, as provided by the Turing machine model.

We can enumerate the programs encoding Turing machines lexicographically by their increasing length (Li and Vitányi 2019, pp. 27-33) and such enumeration contains the shortest program to produce a target object \(x\). This generalizes the unconditional definition as \(K(x) = K(x|\epsilon)\) and is more useful in understanding \(x\)’s structure: if \(K(x|y) \leq K(x)\), then \(x\) and \(y\) share commonalities and \(K(x|y)\) represents the amount of idiosyncratic information left in \(x\).

\(^1\)We use Turing machines with a read-only input tape, one or more (a finite number) work tapes at which the computation takes place, and a write-only output tape. All tapes are one-way infinite, divided into squares, and each square can contain a symbol from a given alphabet \(\{0, 1\}\) or blanks (Li and Vitányi 2019, pp. 27-33)

machine \(U\) terminates with the output \(x\).

\[K(x) = \min_p \{ |p| : U(p) = x \} \]

It is also desired that the UTM \(U\) upon reading exactly \(p\) from its input tape, will read no further and halt with output \(x\). This setup is known as prefix-free coding where no program is a proper prefix of another. It lets us uniquely decode \(p\) to only one object and avoids ambiguous situations where both \(pq\) and \(p\) are valid programs. It also alludes to the algorithmic or universal a priori probability of an object: if the UTM \(U\) terminates with output \(x\) on \(|p|\)-length prefix-free inputs generated by fair coin toss, the probability of \(x\)’s existence, also known as \(x\)’s universal a priori probability is

\[Q_U(x) = \sum_{p: U(p) = x} 2^{-|p|} \]

By being the shortest, \(K(x)\) contributes \(2^{-K(x)}\), larger than any other program, to \(Q_U(x)\) and thus is regarded to be the most probable causal source for the creative product \(x\).

The Kolmogorov Complexity along with all its variants we shall present in this paper are incomputable; there is no function that computes shortest descriptions for all objects. A simple paradigm like: “Supply the UTM with inputs by lexicographically increasing length and the first program that halts with output \(x\) is its shortest program” does not work as it is not decidable which program halts and the computation can go on forever without any meaningful progress. While this might be disappointing, the Kolmogorov Complexity can be reasonably approximated (Vitányi 2020) and its incomputability is perhaps a reminder of the elusive and enigmatic nature of the creative phenomenon.

**Conditional Kolmogorov Complexity, Two-Part Code and Models**

It is often more useful to view one creative product in the light of another, like learning to describe Vermeer’s “Girl with the Red Hat” painting while we already know about his “Girl with a Pearl Earring”. It also illustrates a natural way of describing objects with a predisposition to prior knowledge or inductive bias. We formalize this notion with the conditional Kolmogorov Complexity \(K(x|y)\), which is the shortest program to produce \(x\) on \(U\) when \(U\) is pre-furnished with object \(y\). This generalizes the unconditional definition as \(K(x) = K(x|\epsilon)\) and is more useful in understanding \(x\)’s structure: if \(K(x|y) \leq K(x)\), then \(x\) and \(y\) share commonalities and \(K(x|y)\) represents the amount of idiosyncratic information left in \(x\).

If the provided information \(y\) encodes nonrandom regularities that we recognise and associate with a class or category, then based on the \(K(x|y)\) we can deduce whether or not \(x\) belongs to this class. For example, the “Girl with a Pearl Earring” belongs to a special class of paintings called “Tronie” which features unidentified subjects displaying exotic facial expressions or garments (Schütz 2019). Then the idea of a two-part code lets us decompose the painting into meaningful information— the part that makes it a tronie and individual randomness— the part that separates it from other tronies.
Formally, the shortest effective description of $x$ can be expressed in terms of the length of a two-part code, the first part $K(M)$ describing an appropriate Model computed by a Turing machine $T$ and the second part $K(x|M)$ describing the left-out irregularities or random aspects of $x$ after $M$ squeezes out its regularities:

$$K(x) = \min_M \{K(M) + K(x|M)\}$$

Here Model is a hypernym used to quantify the regularities in sets of objects and with which we recognize the regularities in $x$. The best model $M$ encapsulates the useful or compressible information in $x$, while minimizing the total description length. In relevant research (Vereshchagin and Vitanyi 2004; Gacs et al. 2001), analysis has been mostly done with $M$ denoting a finite set of objects. However, a model can also be a total recursive (Koppel 1995) or probability density function (Gacs et al. 2001), each formulation having its own properties and relevance. To start, let $M$ be $\{x_1,x_2,...,x_m\}$, signifying a history of observed phenomena. The cost of reconstructing an object $x$ from this $M$ on a UTM $U$ comprises of a short program of length $K(M)$ to enumerate the set, while another $\log |M|$ bits to locate $x$ in the set. In this setup, the shortest two-part code $K(M) + \log |M|$ can be larger than $K(x)$, even when $K(M)$ is small. Hence, a complexity restriction is imposed on $M$ : $K(M) \leq \alpha$, $\alpha \in \mathbb{N}\backslash\{0\}$ so $M$ may no longer describe a set losslessly and only captures its essence by exploiting the shared information among the objects. To illustrate this with an example: let $\{x_1,x_2,...,x_m\}$ be different paintings of "crowded field". To transmit one of these through a channel with limited capacity $\alpha$, one can transmit the indication that the painting is of a crowded field and the particular positions of people may be chosen by the receiver at random.

If we formulate $M$ as a total recursive function, the two-part code for $x$ becomes $K(M) + |d|$, where $M(d) = x$ and $K(M)$ is the shortest program length that computes the function $M$. This is a more intuitive interpretation of models as $M$ now can mimic a generator of objects which has embedded in them the structure that $M$ signifies. This is also the foundation of the idea, sophistication which we discuss in more detail in a later section.

Model-fitness and Randomness Deficiency

The finite models $\{M_1,M_2,...\}$ that satisfy the constraints $K(M_i) \leq \alpha$ and $K(x) \leq K(M_i) + \log |M_i| + O(1)$ are called algorithmic sufficient statistics (Gacs et al. 2001). These models allow description of $x$ with only a small increase in complexity, and with a short-to-describe model. The task still remains to choose the one among the candidates $\{M_i\}$ that best-fits $x$ and for that we look at the second element of the two-part code.

Recall that in the event where $M$ denotes a set, it takes about $\log |M|$ bits to locate any $x \in M$. This amount is called the data-to-model code and is different from $K(x|M)$. Since $K(x|M)$ is the smallest program outputting $x$ given $M$, it leverages $M$ and $x$ in a way that could be much smaller than just specifying an index. Going back to the paintings of "crowded field": if the people in $x$ are ordered in a specific way, like a "military parade" then $K(x|M)$ could be much less than $\log |M|$.

![Figure 1: The crowd in figure 1a is more random than 1b](image_url)

In this circumstance, $x$ is not a typical element of $M$. If it were, then any randomly selected painting from $M$ would be indistinguishable from $x$ except for irrelevant details. The difference between $K(x|M)$ and $\log |M|$ is quantified by its randomness deficiency $\delta(x|M) = \log |M| - K(x|M)$.

An object $x$ is typical of $M$ only when we can not significantly improve the conditional description of $x$ given $M$ than specifying its index in $M$, that is $\delta(x|M) \approx 0$. If $M$ is a total recursive function which on some input $d$ generates $x$, then $M$’s fitness for $x$ depends on the number of bits $|d|$ needed to indicate the input. Mathematically, if $K(M) + I_x(M) \leq K(x) + O(1)$ where $I_x(M) = \min \{d : M(d) = x\}$, then the randomness deficiency of $x$ w.r.t. $M$ is $\delta(x|M) = I_x(M) - K(x|M)$.

Typicality and Novelty

We shall now apply the notions of Model and Randomness Deficiency as discussed above to address the “typicality” and "novelty" of a creative product. Ritchie (2007) included “novelty” as one of the essential properties for assessing a product of a computer program exhibiting creativity. But arguing “novelty” or "originality" has an anthropocentric element to them, he added the property of "typicality" to measure the extent to which a produced item is an example of an artifact class. This section refines Ritchie’s argument and formalizes a method for estimating “typicality” of an object. In addition, we discuss how “novelty” can be recognized with the help of a data-to-model code.

Previously, McGregor (2007) addressed this idea by defining the novelty of an object by looking at its information distance (Bennett et al. 1998) from each of a collection of objects of the same class, and choosing the minimum of these distances as the novelty of the new object. In the same paper, the author critiqued the idea by pointing out that the observer estimating novelty needs to have a perceptual frame in which to work. This approach can be fleshed out using the two-part code we use in this paper, and also by considering how computational agents update their perceptual model of the object by interacting with each other and with new artifacts; this latter subject is one we are currently exploring.

The question of how typical a creative object $x$ is, with respect to a composer’s oeuvre or a even broader artifact class, is really understood by its randomness deficiency $\delta(x|M)$. Here $M$ models the apriori regularities or recognized prop-
properties of a representative set $S = \{x_1, x_2, \ldots, x_n\}$, which, following Ritchie (2007) we call the “inspiring set”. We express $M$ as a total recursive function such that there exist parameters $\{d_i\} : M(d_i) = x_i$ for $1 \leq i \leq m$. Let $l_x(M)$ denote the length of the first parameter $d$ on which $M$ halts with the artifact $x$. Thus as $\delta(x|M) = l_x(M) - K(x|M)$ decreases, $x$ becomes more typical for $M$. Note that based on $S$, the typicality that $x$ exhibits can be akin to either $H$-creativity (producing an idea/artifact which is wholly novel within the culture, not just the creator’s oeuvre) or $P$-creativity (producing an idea/artifact which is original as far as the creator is concerned) (Boden 1991). However the analysis we present here is effective for understanding both, if $S$ contains members that are consistent with the class that we are interested in. Then we have the following definition of typicality.

**Definition 1** Let $M$ be a total recursive function and a minimal sufficient model of the inspiring set $S = \{x_1, x_2, \ldots, x_m\}$ such that there exist parameters $\{d_i\}$: $M(d_i) = x_i$ and $K(M) + |d_i| \leq K(x_i) + O(1)$ for $1 \leq i \leq m$. Then the typicality of an object $x$ with respect to this model is as following.

$$\text{typicality}(x|M) = -\delta(x|M) = K(x|M) - l_x(M)$$

When $x$ is not in $M$’s range, that is $\forall d : M(d) \neq x$, then $l_x(M) = \infty$ and we get the lowest typicality $-\infty$. If typicality is close to the maximum value 0, then there are no simple special properties that single $x$ out from the majority of elements in $S$. Otherwise, we can pick a special subset $Q$ of $M$, which has only the members with this property (like ordered positions of people in the “crowded field”), then $x$ will be much more typical for $Q$, than it is for $M$.

Intuitively, a “novel” object with respect to a model should have high randomness-deficiency. But this property alone is not sufficient, as to define “novelty” we need a notion of unexpected or unique outcome of the corresponding model. The central motivation in our discussion has been to find the true source $M$ that produced the object at hand. But suppose the true source is 100 coin flips and our data is 1111 . . . 1. A model that identifies with flipping a fair coin as the cause of the data, is surely a bad model. However, in real-world problems, such as modeling creative products, the data can be just atypical or accidental for the model that actually produced it. In this case, the model might still describe the regularities in the object, but the extraordinary conditions or the data-to-model code that caused the model to output it, sets the object apart from other objects created by the same model. We liken a novel artifact to such data: it is an unlikely and original outcome of a model $M$ that is a minimum sufficient statistic for the inspiring set $\{x_1, \ldots, x_m\}$ and there exist parameters $d_i, d : M(d_i) = x_i, K(M) + |d_i| \leq K(x_i) + O(1)$ for $1 \leq i \leq m$ and $M(d) = x$. The artifact $x$ is producible from model $M$, but the remaining distinctiveness $d$ in $x$ is an indicator that there are better models to produce $x$ (consider the model $M$ accidentally getting a set of ordered locations to place people in the “crowded field” example when location are being randomly generated). This unfitness is captured by its reduced kinship with other artifacts $\{x_i\}$ for which $M$ is a model. Since, these objects are results of inputs $\{d_i\}$ and $d$ to the model $M$ which is the common denominator between the inspiring set and $x$, the extent to which $x$ is novel is determined by how much information in shared between $\{d_i\}$ and $d$. Novelty of $x$ is then essentially captured by the mutual information $I(\{d_i\} : d) = K(d) - K(d|\{d_i\})$ between $d$ and $\{d_i\}$ (Li and Vitányi 2019, p. 249). The less $\{d_i\}$ informs of $d$, the more novel $x$ is, reaching maximum at $I(\{d_i\} : d) = 0$. This could implicitly mean that the randomness deficiency $\delta(x|M)$ is large. But $I(\{d_i\} : d)$ measures the difficulty with which we figure this unfitness out even with the available information.

**Definition 2** Let $M$ be a total recursive function and a minimal sufficient model of the inspiring set $S = \{x_1, x_2, \ldots, x_m\}$ such that there exist parameters $\{d_i\}$: $M(d_i) = x_i$ and $K(M) + |d_i| \leq K(x_i) + O(1)$ for $1 \leq i \leq m$. Then the novelty of an artifact $x$, producible from $M$ with parameter $d : M(d) = x$, is

$$\text{novelty}(x|M) = -I(\{d_i\} : d) = K(d|\{d_i\}) - K(d)$$

Novelty thus can be compared to explorations undertaken by an artist or changes in a genre while not affecting their creative styles: learning and applying new techniques, exposure to new environments and influences, can bring about novel objects that capture the general spirit of the inspiring set but are difficult to be recognized in their light.

It is important to note here the distinction between a two-part-code (model and inputs) identified by an observer and the actual computation process followed by a creator. The observer may be updating an already-existing model based on new examples, or may in fact not be capable of building a computation structure as sophisticated as the one the creator used; for example, for a logically deep object (defined below as an object needing long computation by a short program), the observer may not even have time to run all of the steps the creator used, while updating or building a two-part code for the object. This takes us to the discussion of time-bounded complexity analysis and modelling the creator, which we discuss in detail in the next few sections.

**Logical and Computational Depth**

We now take into account the difficulty or resource with which $p$ outputs $x$ or transforms another object $y$ into $x$. Indeed, the shortest program $p^{\text{sh}}$ which computes some object $x$ in a bounded time $t$ and space $s$ can be significantly larger than the shortest program $p$ that has access to unlimited resources. Note that, programs using restricted time are more interesting to analyse than programs with restricted (polynomial) space but unlimited time, as they still can solve the hardest of problems (think about the program that generates the core ideas in this paper with only 8-pages to work with) (Li and Vitányi 2019, pp. 37-39), and it also helps quantify the number of steps taken by the program. Hence, we focus on programs $p^t$ that generate a target object $x$ within a bounded time $t$, while not being too inefficient in their use of space. The time-bounded Kolmogorov complexity $K^t(x)$ is then defined by

$$K^t(x) = \min_{p} \{ |p| : U(p) = x \text{ in at most } t \text{ steps.} \}$$
If \( t \) is a shorter time span than the original time taken by the shortest program for \( x \), then \( K_t(x) \) suffers from \( K_t(x) \) redundancy or adhocness, that is \( K_t(x) \) may have to store some information about \( x \) without meaningfully compressing it. This difference is known as the Computational Depth of \( x \) (Antunes et al. 2006).

\[
cdepth_t(x) = K^t(x) - K(x)
\]

As \( t \) grows, excluding the pathological cases (programs doing unnecessary computations), the non-randomness in \( x \) gets disguised by complicated manipulations or computations by the program. Bennett (1988) thus calls the time taken by the shortest of programs for producing \( x \) its Logical Depth.

\[
ldepth_t = \min \{\text{time}(p) : U(p) = x \text{ and } |p| \leq K(x) + b\}
\]

The minimum is taken among the available candidates to avoid selecting programs that despite producing the desired object, do so inefficiently, whereas other similar-length programs are faster. The term \( b \), called the significance level of \( ldepth_t(x) \), calibrates the added length and assigns an importance or confidence measure to the program: as \( b \) gets smaller, the program that witnesses \( ldepth_t(x) \) becomes more likely to be the actual program that generates \( x \). Thus the number of steps taken by this program is equally probable to be the time needed by \( x \) to evolve from its short description.

**Value as Computational Effort**

We now make a case for Logical Depth as a formal measure of value. We propose that what makes a creative object valuable is not its information content, but rather the amount of mathematical or creative work it relieves its receiver from repeating, which was plausibly done by its originator. A sequence that represents the outcomes of \( n \) coin tosses, has high information content but little value. Conversely, a book on algebra may list a number of difficult theorems, but has very low Kolmogorov Complexity since all the theorems are derivable from the initial few definitions and axioms. However, such derivations can be time-consuming and if we transmit only a short description containing the theorems of the book, a receiver has to spend a long time to reconstruct their proofs. Sending the entire book does not increase the information content transmitted, but now the receiver has all the useful information readily available. Thus value of an object does not depend on its absolutely unpredictable parts (information content), nor on its obvious redundancy (verbatim repetitions, sequence of 1’s), but rather on what might be called its buried redundancy—parts reproducible only with difficulty, things the receiver could in principle have figured out on their own, but only at considerable cost in resources or computation (Bennett 1988).

This approach to value is obviously only about the object itself, not about its cultural significance, its ability to be understood by viewers, or any other social properties. Yet, despite this limitation, we are not the first authors to make this connection between the value of an object and its buried computational value; Vidal and Delahaye (2019), in particular, has cited exactly this same quantity in their proposal of an ethical mandate to protect artifacts that contain computational significance of the same sort.

A delightful example of a logically deep object is the characteristic sequence of the diagonal halting problem, \( \chi \), where each bit \( \chi[i] \) is 1 iff the \( i \)th program halts. Despite its apparent importance, the \( n \)-bit prefix \( \chi_n = \chi[0 \ldots n - 1] \) of \( \chi \) is highly redundant with \( K(\chi_n) = \log n + O(1) \). The intuition is we only need to specify the number of indices that contain 1. Once this \( \log n \) number is known we can dovetail all the programs \( p_0, \ldots, p_{n-1} \) (Li and Vitányi 2019, p. 181) on an UTM and stop the computation once the desired number of programs have halted. Yet, this is computationally very expensive, taking at least as much time as the slowest program in the above enumeration.

A logically shallow object, on the other hand, has a fast-running program that is highly probable to be its source. Note that, defining logical depth as the runtime of the shortest program \( x^* \) does not constitute a stable definition since there might be a program of just a few more bits using substantially less time to generate \( x \). A complex artwork may have a slow-running short program that is not much shorter than the print program that outputs the artwork literally.

\[
K(x) \leq |p\text{\_print}(x)| + n + O(1)
\]

Here, the object lacks internal redundancy that could be exploited to encode them concisely and is logically shallow, as a print program which generates the object quickly, is almost as probable to be its origin as the shortest one. In a contrasting scenario: a painting that appears complex to its observer may have a short program to generate it. Colton (2008)’s “Art Exhibition: Dots 2008” describes a similar case where a painting of some random dots on the canvas is given two plausible explanations. One, “the dots are randomly arranged” and two, “the dots represent some friends of the artist and the colors and positions convey the artist’s feelings about them”. To a spectator, who has had a bounded time \( t \) to analyse the painting, the first explanation may seem most plausible in the form of time-bounded Kolmogorov complexity, \( K^t(x) \). But knowing that there is even a shorter, if slower to execute description of the painting, \( \text{Partist} : K(x) \leq |\text{Partist}| + O(1) \) available, the spectator is more inclined to accept \( \text{Partist} \) as its most probable origin and assign to it a value as the computational effort \( \text{time}(\text{Partist}) \) suggests. Thus:

**Definition 3** The value or quality of a creative product \( x \) is the minimum computational effort or time needed to produce it from a \( b \)-significant shortest description.

\[
\text{value}_b(x) = \{\text{time}(p) : U(p) = x \text{ and } |p| \leq K(x) + b\}
\]

However, as we write about computation that is constrained by runtime or space, it is also worth considering certain complex artifacts that may have a short description but the only way to reproduce them from the description would require unbounded resources. As such, there is no other way to specify them than to spell them out bit by bit and if a creator were to claim the existence of such an artifact, no effective programmatic verification of this fact would be possible. The situation for such objects, as with Colton’s Dots 2008...
exhibit, is complicated: a Turing machine that knew about the personal relationships in Colton’s artist’s life might be able to much more sharply compress those paintings. But without the access to one, the observer is burdened with the verification of the artist’s claim.

The Creative Process and its Non-randomness

We inevitably arrive at the questions of what constitutes an effective generation process for an artifact and in the presence of multiple plausible theories, which process is the most likely to have occurred. A fraudster or charlatan may claim that a complex-looking creative product is a result of a slow-running short program, thus artificially inflating its value. But, if the object has an equal-length fast program which involves no random steps in generating the object, then it is equally likely that the latter is the true generative process for the object; and if the object can be produced by taking a small number of non-random steps, then it is certainly possible that the fraudster program takes unnecessary pathological steps in order to seem serious.

This non-random non-trivial effort to generate an artifact is also stored in its subjective organization: Beethoven’s “Für Elise” is aesthetically pleasing because it is organized in a certain way. If we rearrange its notes randomly to make different musical pieces, only a handful among the vast majority of resulting pieces will be deemed musically pleasing and perhaps only one will be as musically valuable as “Für Elise”. Similar idea appears in the fiction of Borges (1998), “The Library of Babel”, which describes a library all possible 410-page books on a 25-letter alphabet, and the librarians’ attempts to discern which of the books were meaningful. Thus, the inherent organization of these products hold clue to the non-trivial and laborious processes that resulted in their existence (Bennett 1988).

Hence, the most plausible creative process is carried out by a program that is no more than $b$ bits longer than the shortest and from which the work to reproduce $x$ involves no unnecessary, ad-hoc assumptions except for the $b$ bit redundancy. This necessary and non-random workflow for the production of $x$ is what we assign credence to (recall the preference of the art-lover in Colton 2008’s dots exhibit). If a short program $p$ has a slow deductive reasoning process, it is not evidence against the plausibility of this program. In fact, if the product has no comparably concise programs to compute it quickly, it is evidence of the non-triviality of the generative process. A great work of autobiography is one example of this: if we just consider the written text as its acceptable representation, then its information content is really low (Shannon 1951). But the existence of such literature stands evidence of a profoundly-led life by the author and the significance of the events that happened in that lifetime.

Definition 4 A b-significant creative process of a product $x$ is simulated by the UTM upon input $p$ such that $|p| \leq K(x) + b$, $U(p) = x$ and $p$ takes the minimal non-random steps among all b-incompressible programs for $x$.

Is it possible to convert a shallow object like a random string to something deep like the Tolstoy’s “War and Peace”? Satisfyingly, this is answered in the negative by the slow growth property: a fast deterministic process is unable to transform a shallow object into a deep one, and that fast probabilistic processes can do so only with small probability (Bennett 1988).

Sophistication

We formalize a notion of “structure” or “projectable properties” in an object. In the previous sections, we have mentioned that the complexity and usable information in an object do not have a causal relationship; in fact they may very well be orthogonal properties of an object (Koppel 1995). We thus try to decouple the part of an object that is an aggregate of shareable properties from its accidental information with a two-part code. Earlier, when discussing models, we introduced total recursive function as a way of describing a set of objects with respect to which we examined $x$’s regularities. Here we use a total recursive function slightly differently, to capture the structural information in an individual object $x$ that shows evidence of some planning that went into $x$’s generation. The utility of such formulation becomes evident through the following example: consider a total function $double(x)$ that on any input doubles the bits, $double(011) = 001111$, $double(101) = 110011$. Such properties are difficult to find in general, but work as an excellent compression scheme for the product. Hence the sophisticated part of an object is the size of such programs which stands for the non-random structure of an object.

The e-sophistication of an object is thus defined as follows (Koppel 1995).

$$soph_e(x) = \min\{|p| : p \text{ is total, a parameterization } d \text{ exists for which } U(p,d) = x \text{ and } |p| + |d| \leq K(x) + c\}$$

That is, $x$ is sophisticated if the best model for $x$ is a comparatively long program, not something as simple as the double function. The size of an optimal total recursive function along with the data may be $c$ longer than $x$’s shortest description. But in order to reduce $c$ if we furnish $p$ with properties that are accidental or exclusive to $x$, then it might fail to recognize objects that are similar or generated by the same source. Mondol (2020) demonstrated an example in symbolic-music-compression using context-free patterns. In order to capture core-properties of a music-piece, the author only considers the patterns that repeat most in the corresponding composer’s oeuvre. This way the emphasis is put more on the inherent composition techniques of the sequence rather its distinct embellishments. The significance parameter $c$ is thus interpreted as a confirmation of the description $(p,d)$ before regarding extra structure represented by a longer program $p'$.

Attributes of the Creator

Sophistication is a natural way to measure how much information of an object we can throw away without losing the ability to query its properties (without false positives). We propose that such properties are also a measure of the creator’s attributes embedded into the product’s structure. Suppose that a music generator whose inner mechanism is unknown to the observer is broadcasting self-composed music.
If the composition obeys some simple rule such as repeating the same patterns or sounds maximally random, then we would not attribute craftsmanship to the source. However, the composition exhibits complex structure, which is only possible through rigorous planning and meaningful exploration, we might suspect the existence of a skilled creator. Hence, sophistication is that quality of an object that sets apart the artist’s talent from their fanciful impulses.

**Definition 5** The style or signature of a creator inherent in a creative product $x$ is measured by the object’s c-significant sophistication $\text{sofh}_c(x)$. That is, we can say that the generator program $p$ of length $\text{sofh}_c(x)$ produced $x$ on input $d$ leaving out all but $c$ bits of redundancy.

It is worthwhile to note that both logical depth and sophistication are measures of meaningful complexity in an object; but while one uses dynamic resources (program time), the other uses static ones (program size). Thus the two measures are not necessarily correlated (Antunes and Fortnow 2009): the halting sequence $\chi$ is logically deep, but has low sophistication ($O(\log n)$). Rather, logical depth can be used as a structure finding mechanism. We formalize this with the converging hypotheses argument (Koppel 1995): consider the same music generator as above; as we observe more of its music, more structure becomes apparent, thus we exhaustively search through all programs (not necessarily total) $p$ : $|p| \leq n$ and data $d : |d| \leq n - |p|$ in order of increasing length and let them run for $\text{ldeth}_c(x^n)$ steps. We choose the shortest $p = p_n : |p| + |d| \leq K(x^n) + c$ that satisfies these criteria. Thus, in our model, as more composed music is observed, previous hypothosized generators are abandoned for one of two reasons. The most straightforward reason is that the subsequent parts of the composition is inconsistent (do not fall in the generator’s range). In this case, the program is changed in favor of one which is less powerful (shorter, using longer data). The other reason for abandoning a program is that as more parts of the composition is observed, structure becomes apparent which was not previously so- that is, use of a more powerful, longer program results in a shorter description when including the required input to generate $x^n$. Such procedure might not give the smallest compression program for the music generator itself, but it increasingly describes the properties of initial segments of its generated music $x$, which can be used to compress the larger initial segments increasingly better as $n \to \infty$.

**Non stochastic Objects (Masterpieces)**

Finally, we discuss another remarkable outcome of the notion of sophistication: absolutely non-stochastic objects, whose complexity is mostly comprised of non-random structure (high sophistication and useful properties) and show that creative masterpieces fall into this category.

An absolutely non-stochastic object has neither minimal nor maximal complexity. Hence they are not typical outcomes of any total recursive program that exhibits low structure. Additionally, non-stochastic objects have no optimal programs that are of relatively small complexity; that is they exhibit high randomness deficiency or atypicality for a program $p : K(p) < K(x)$. Rather, these objects are typical outputs only of programs $p$ that have complexity close to their own, $K(p) \geq K(x)$, indicating high sophistication (Gacs et al. 2001). The program part $p$ of such an artifact thus showcases the creator’s elaborate techniques, contents and creative properties that can be pioneering and replicated. When a non-stochastic object is an output of a unique highly sophisticated program, it depicts innovation; similar to the transformative effects of a masterpiece, it has the ability to push a medium or genre to new directions. Thus absolute non-stochasticity is a pre-cursor to creative masterpieces.

**Definition 6** A creative masterpiece $x$ is absolutely non-stochastic or highly sophisticated, that is, they exhibit low randomness deficiency (needing small additional data) only for total recursive programs $p$ that have complexity close to their own, $K(p) \geq K(x)$. For programs $p$ with $K(p) < K(x)$ they will either require larger additional data or they will not be in those programs’ range at all.

If we were to partition a creative masterpiece into meaningful complexity and random noise, we will find that almost all of its complexity comes from useful incompressible properties. Moreover, the amount of such non-randomness is also significant. Thus non-stochastic artifacts reside in a Goldilocks complexity zone: Shen (1983) showed that these objects have complexity at least $K(x) \geq \frac{n}{2} - O(\log n)$. Thus a masterpiece, which is a product of its generative program $p$ having relatively high complexity while being absolutely non-random, is an extremely rare phenomenon.

**Re-examining Computational Aesthetics**

With the computational tools we now have available in our armament, we revisit some of the existing attempts to incorporate algorithmic information theory into evaluating creativity. Among the more recent papers, Ens and Pasquier (2018) demonstrated a way to evaluate style imitation systems by comparing their generated artifacts with the reference artifacts. They analyse the statistical significance of inter-corpora artifact distance which is approximated with the normalized compression distance (NCD) $\frac{K(xy) - \min\{K(x),K(y)\}}{\max\{K(x),K(y)\}}$. The NCD is a very natural approach to measure how different two artifacts are (the numerator is the amount of information that $x$ and $y$ differ by), but as we have seen in the model section: when multiple artifacts are concerned, a more theoretically correct way is to contemplate a model that describes them well. Then two corpora are maximally different if their individual artifacts exhibit high randomness deficiency w.r.t. each other’s model.

Another school of thinkers (Birkhoff 1933; Moles 1966) expresses aesthetic beauty as a ratio of order and complexity $\frac{O}{C}$. Moles (1966) approximates this ratio with relative
redundancy $|x| - K(x)$ equating order with $|x| - K(x)$, the amount by which $x$ can be compressed. While this might work for objects that lie at either extreme of randomness (strings formed by coin-toss or repetitions of bits), it fails precisely for objects that are in-between: e.g. for a highly sophisticated object this definition measures exactly the opposite of order, as such objects exhibit structure through almost all of their complexity. Kosheleva et al. (1998) propose quite a different way of approximating $O(x)$ altogether. They equate complexity $C$ with the time $time(p)$ a program takes and order with $2^{-l(p)}$. While the definition of order is similar to the former approach (both argue that the smaller $l(p) = K(x)$, the more order $2^{-l(p)}$ or $x - K(x)$ is there in the object), we now know that $time(p)$ is far from being a measure of complexity of an object; rather it marks the evolution-time of an object from a short program.

Schmidhuber (1997) has also given a framework for “beauty”, a concept we are not considering much in our paper. His argument is that for each observer (which in our frame we treat as a model, of the total function variety), the most beautiful objects are the highest-probability objects. This suggests that the most beautiful objects are also the most typical ones, which is worrisome, since most models also have low sophistication (and hence, the most “beautiful” objects will be comparatively trivial). Schmidhuber’s approach also does not allow easily for models to be adapted in light of new objects, nor in light of new explanations highlighting the complexity of existing objects.

A Formal Framework We can reformulate the aesthetic beauty ratio $O(x)$ of an artifact $x$ by defining order with $K(M)$, where $M$ models the regularities in $x$ leaving out $K(x|M)$ randomness and $K(M) + K(x|M) \leq K(x) + O(1)$. Based on our discussion, $M$ can be built either from a coherent set of objects with which we want to recognize $x$ or it can be its sophistication or useful properties that can be used to generate similar objects. The denominator or complexity is simply the raw Kolmogorov complexity $K(x)$. Then $K(M) / K(x)$ assigns highest aesthetic beauty to masterpieces and lowest to objects that exhibit low structure (random strings or sequences of 1’s have low complexity models: fair coin-toss generators or printing $|x|$ 1’s). Thus, in general, the algorithmic recipe presented in Table 1 can be followed for aesthetic analysis of computational creativity. We note the relationship between the entities in the first column of the table and three of the four P’s of creativity (Jordonous 2016; Rhodes 1961); in a future work, we will integrate the fourth P (press) through analysis of criticism as a creative task in its own right.

<table>
<thead>
<tr>
<th>Creative Entity</th>
<th>Attributes</th>
<th>Algorithmic Information Theory Notion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Artifact</td>
<td>Typicality</td>
<td>Randomness Deficiency</td>
</tr>
<tr>
<td></td>
<td>Novelty</td>
<td>Mutual Information between model parameters</td>
</tr>
<tr>
<td>Order and Noise</td>
<td>Non-randomness</td>
<td>Effective b-significant program</td>
</tr>
<tr>
<td>Creative Process</td>
<td>Value (also of artifact)</td>
<td>b-significant Logical Depth</td>
</tr>
<tr>
<td>Creator</td>
<td>Skills and Style</td>
<td>Sophistication</td>
</tr>
<tr>
<td></td>
<td>Masterpiece</td>
<td>Non-stochasticity</td>
</tr>
</tbody>
</table>

Table 1: An algorithmic recipe for Computational Creativity

an artifact’s actual and apparent complexities: an observer or critic’s time-bound explanation $K^t(x)$ of an artwork $x$ can be influenced by the real creative process of the artist; while they can also dismiss the fraudulent claims of a charlatan by seeing through the actual value of an artifact. Although this aspect of creativity will be expanded on more in a future paper, such interplay between artists and critics has been often absent from previous computational understandings of creative work. Additionally, the notion of sophistication lets us illustrate a creator’s virtuosity present in their creative product. The input $d$ to a generator program $p$ that we called accidental information, can be thought of as the inspiration or an encoding of the surrounding environment that influences a creator program $p$. A particular delightful outcome of sophistication is its ability to describe masterpieces in the form of highly sophisticated artifacts, whose existence is only possible through a similarly sophisticated program. Although these concepts are not computable, they provide a reliable theoretical foundation upon which other models (e.g. machine learning) can be built and evaluated.

Acknowledgement
The work of Dan Brown is supported by the Natural Sciences and Engineering Research Council of Canada. We are very grateful to the anonymous reviewers for a number of helpful suggestions in their thorough reviews.

References


Computational Creativity and Consciousness: Framing, Fiction and Fraud

Geraint A. Wiggins
AI Lab, Vrije Universiteit Brussel, Belgium & EECS, Queen Mary University of London, UK
geraint@ai.vub.ac.be

Abstract
Computational Creativity, like its parent, Artificial Intelligence, suffers from ill-definition: both “creativity” and “intelligence” are difficult, perhaps even impossible, to define. Both fields have also suffered from confusion about the relationship between their key concept and the equally problematic concept of “consciousness”. Computational Creativity, however, has yet to address this issue effectively, which can only be detrimental to the field. This paper attempts to lay out the issues, to identify useful boundaries, particularly with respect to framing and the generation of meaning, and to map out ways in which Computational Creativity research may navigate the landscape of scientific possibility, while remaining true to itself.

Overview
In the current paper, I attempt to engage with some open questions regarding the relation of creativity, and specifically computational creativity, to consciousness and meaning. While definitive answers are not yet achievable, I suggest that the discussion is progressive and useful. In this context, I discuss computational creativity research methods, with particular emphasis on the notion of framing (Pease and Colton, 2011). First, relevant aspects of the philosophy of artificial intelligence (AI) are summarised, pointing out a new interpretation. Then, aspects of research on consciousness and its relation to AI are reviewed. While most of the ideas presented in these first two sections are probably familiar to many readers, they supply a specific context for what follows. Next, the relevance to computational creativity, practical and theoretical, of these issues, is discussed, in context of recent publications in the field. Finally, some principles are proposed that may help computational creativity research to make progress as a scientific endeavour.

In summary, the conclusions argued in this paper are as follows:

1. Computational creativity, as a scientific discipline, should (and mostly does) primarily focus on identifying those elements of creative systems that are necessary to allow creativity to emerge from their operation. Thus, computational creativity can predominantly be viewed as an ex post phenomenon emergent from computational systems.

2. Any attempt to introduce aesthetics into computationally creative systems must account for the origin of the aesthetic so introduced in a philosophically sound way.

3. In order to demonstrate scientifically that computational systems enjoy such an ex post capacity as creativity, evaluation methods used must be honest, in the sense that any detected emergence of creativity should be, whether directly or indirectly, explicitly attributable to the processes within those systems. In particular, when using metaphor to explain outcomes from computational creativity (or any other research field), it is important to be explicit about the metaphorical nature of the explanation.

4. This notion of honesty must extend into the application of framing to computationally creative systems and their outputs.

Context: AI, Creativity, and Consciousness
In his seminal paper, Computing Machinery and Intelligence, Alan Turing (1950) sets out to address the question “Can machines think?” He begins from the position that, stated in this way, the question is “too meaningless to deserve discussion” (p. 442), because the words “machine” and “think” are ill-defined. To address this problem, he defines precisely what he means by “machine”—a digital computer—and proposes a thought experiment to help understand the question, the Imitation Game, which has come to be known as “The Turing Test”. We are invited to consider the question of whether a machine can convince a human that it is a human, in competition with a human, who is attempting the same, via a typed dialogue.1

The status of “intelligence” in Turing’s extraordinary paper is never made specific. Aside from the title, the word occurs only once, explicitly in the context of human intelligence, and the connection between “can…think” and “is intelligent” is left for the reader to assume. The word “intellect”, also, is used only to describe human capacity. So Turing is proposing a proxy for the question, “can machines think?”, which avoids the question of what it means to “think”, by substituting a comparison with something (a healthy adult human) that is generally agreed to be able to

---

1When people appeal to the “The Turing Test” in the modern scientific literature, it is usually to a less challenging form, in which there is no opponent, and often even no dialogue.
think. Many definitions of Artificial Intelligence are similarly couched in terms of “things a human can or might do” (e.g., Bellman, 1978; Haugeland, 1985; Charniak and McDermott, 1985; Winston, 1992), and in some definitions of computational creativity (e.g., Wiggins, 2006a).

Turing addresses various objections to his replacement of that original question that are not relevant to the current discussion: religious dualism, pro-human and anti-machine arguments of various degrees and kinds, and formal mathematical arguments. Two further counter-arguments, that are relevant here, are what Turing calls the “argument from consciousness” (p. 445) and (charmingly) “Lady Lovelace’s Objection” (p. 450). These are addressed in later sections.

It is noteworthy that Turing never explicitly suggests that the Imitation Game should be used as a scientific test for intelligence in a given computer—instead, he explicitly posits it as an alternative to the general question about “imaginable” machines (p. 436), and not about a particular machine. He refers to the (computational) Imitation Game as a “test” only three times. Nevertheless, the paper is widely supposed to be a specific proposal for a functional test to identify the presence of intelligence. An alternative view of Turing’s argument, since he also does not claim that a computer is like a human brain, is as follows. Turing could be presenting a thought experiment, making the point that, without a functional definition of “thinking”, one cannot distinguish “real thinking”, done by a person, from the outward appearance of “thinking”, by an adequately programmed machine, even if what the machine is doing is not “real thinking” at all. The inevitable conclusion from this perspective is: if the behaviour of the machine appears close enough to behaviour arising from human “thinking”, one can no longer tell whether it is “really thinking” or not. This argument bears comparison with the “process vs. product” argument in creativity studies, and the answer is the same: for a meaningful comparison between human and artificial, both process (i.e., mechanism) and product are important.

This interpretation is not the widely accepted intent of Turing’s paper. However, whether or not it is his intended interpretation is immaterial for the purposes of the current argument. For, whether Turing intended it or not, the above is indeed a valid consequence of his argument: if we define intelligence only by our ability to recognise its effects, we can be fooled. In principle, a sufficiently detailed, but nevertheless stateless (as defined by Russell and Norvig, 1995), agent, A, will be enough to fool us.

This is a version of the Chinese Room argument (Searle, 1980), to which we return below.

In summary, the best definition available of intelligence as a property of an agent renders that property detectable only by observation of behaviour of that agent in a given context: a firmly ex post definition. Therefore, two healthy adult humans, sitting, motionless, side-by-side are utterly inscrutable and indistinguishable with respect to that property, no matter who they are.

Russell and Norvig (1995), who currently inform students worldwide, effectively circumvent this problem by defining it away: for them, artificial intelligence should exhibit rational behaviour, which is in turn defined in terms of actions that lead to a goal. Thus, artificial intelligence is rational agency, and an agent that does not have goals (like the stateless agent, A, above) is by definition not intelligent, reversing the logical order to ex ante. While this definition works well from the practical engineering perspective of getting at least something done, it quickly becomes clear that it is incomplete, because intelligent organisms do more than just attain goals—they also identify and formulate them. Thus, Turing’s “thinking” machines are reduced to reactive (albeit flexible) slaves whose goals are dictated, presumably by proactive humans. Furthermore, this particular definition of “rationality” excludes what, in a human, would be considered emotion or affect—that is, the aesthetic side of intelligence. In defence of Russell and Norvig (1995): one has to start somewhere.

Lady Lovelace’s Objection Ada, Countess of Lovelace was arguably the first software analyst, studying the output from Babbage’s computing engines in the first half of the 19th Century. Startlingly ahead of her time, she considered the possibility of computational creativity, but concluded that a machine was not capable of creating something new, because it could only do what its program(mer) told it to do (Countess of Lovelace, 1842). Computational creativity researchers remain under siege by this counter-argument, 200 years later: the Objection has been resurrected at every single public talk on computational creativity that the current author has given in the past 20 years. Turing (1950, p. 450), equally startlingly ahead of his time, refutes it with an argument that still holds: if a machine can learn, and base its productions on what it has learned, then there is no reason why, in principle, those productions cannot be novel, and independent from the programmer. This is a large step towards autonomous creativity.

The Argument from Consciousness Turing responds to “The Argument from Consciousness”, made by Jefferson (1949), that a machine, capable of “real thinking”, should be able to perform creative acts because of the thoughts and emotions that it feels, and also know that it performed them. Thus, human-level self-awareness is invoked, in what is often classed as the highest level of consciousness (Merker, 2007, and below): experiencing not only one’s existence, but also awareness of one’s own existence; experiencing knowing that one exists; knowing that one is capable of action in the world, and so on. In his rebuttal, Turing does not draw a clear line between this construct and what he means by intelligence. He argues that if consciousness is a necessary part of thinking, then the only way to demonstrate true thinking by a machine would be to be the machine and to experience the thinking, first hand: the solipsist position. We avoid the solipsist position in human society, Turing charmingly notes, by agreeing “the polite convention that everyone

---

2 Though he did, in fact, colloquially refer to Universal Turing Machines as “Brains” (Hodges, 1992).

3 A reasonable rebuttal here is that such an agent would be impossible to build, because there would be too many cases to include in its production list. However, for the purposes of the current argument, the theoretical possibility suffices, because, no matter how complex the world, the Imitation Game endures only for finite time.
thinks” (Turing, 1950, p. 447). Thus, the answer seems to be “this question is ineffable” and that people will yield before being cornered into solipsism. This is a rhetorical argument, not a logical one, and so it does not satisfy.

More usefully, Turing confirms that he does see the “mystery about consciousness” (Turing, 1950, p. 447), and denies the need to solve that mystery before answering his central question. In doing so, he implies that consciousness is something (at least partly) separable from intelligence, and this is more telling than his actual rebuttal: if intelligence can be reproduced in a machine without consciousness, then intelligence sits far more comfortably in what Turing explicitly defined as a machine than otherwise.

The Chinese Room Perhaps surprisingly, since Turing defuses this issue explicitly in his paper (see above), the Chinese Room began as an argument against what Searle (1980) called “strong AI”: that a machine could have a mind in exactly the same sense that (we all politely assume) humans do. This definition tacitly conflates the property of intelligence with the property of consciousness, and these two are not the same (Turing, 1950; Preston and Bishop, 2002). Briefly summarised, the argument posits a closed room with a person inside, who does not speak or read Chinese. That person has access to a large body of knowledge, expressed in terms of Chinese characters. This knowledge is presented in such a way that it may be deployed by having the person match the characters together, without understanding their meaning. Thus, questions, posed in Chinese, and posted on paper through a hatch in the wall, maybe answered, also in Chinese, by the person matching and copying the relevant characters, and passing them out on paper through another hatch. Thus, the room appears to answer questions, but it cannot be said to understand the questions and answers in the sense that the person within would understand them in the person’s own language. Searle (1999) changed his position on this, acknowledging that the Chinese Room is not so much an argument against artificial intelligence as against machine consciousness, and in this context it is indeed more successful: few artificial intelligence researchers claim to be developing human-like consciousness in their machines. Indeed, such an attempt would be ethically questionable, because, if it were successful, the “off” switch would become a murder weapon.

Context: Conscious Experience and Aesthetics

Previously, we rehearsed the Chinese Room argument regarding consciousness in computers. We now address the concept of conscious experience in general, as discussed by many philosophers and others (e.g. Wittgenstein, 1958; Nagel, 1974, 1986; Searle, 1980, 1999; Dennett, 1991; Merker, 2007, 2013; Shanahan, 2010). Although we are ultimately interested in the relationship between computational creativity and consciousness, we begin with the thought experiment of Nagel (1974): “What is it like to be a bat?”

First, it is important to understand that Nagel’s usage of “What is it like to be…?” is more specific than the everyday English usage. The author of the current paper, a university professor, could imagine what it is like to be, for example, a politician, thinking through the visible aspects of a politician’s life, and transferring human perspectives between life experiences, real and imagined. This is the everyday usage: the professor is imagining what the politician’s life is like from her own perspective. Nagel’s usage is more specific: he refers to the bat’s experience of its own existence, explicitly ruling out the experience of a human pretending to be a bat; this will be significant later in our argument. It is like something to be the current author: it is like something else to be another human, the reader perhaps, though experience suggests that there are commonalities which can be conventionally agreed via reference to the external world; it is like something else again to be a bat, and there are aspects of bat experience that are not communicable to humans, and therefore could not be conventionally agreed, even if we could discuss them.

For the purpose of the current discussion, it is helpful to carry Nagel’s argument further. The reader is now invited to try to imagine, in Nagel’s sense, what it is like to be a rock. Even though the behaviour of rocks is significantly more complicated than that of bats, it is impossible for a human to imagine what it is like to be a rock. This is not just because of the significant physical differences between rocks and humans, but because, in Nagel’s sense, it is not like anything to be a rock. We cannot meaningfully say “from the rock’s perspective,” because the rock has no perspective. We cannot imagine what it is like to be a rock, from the rock’s perspective, any more than we can imagine our own future experience of being dead.

Having established an experiential boundary between the rock (“not like anything”) and higher biological species (“like something”), we skip discussion of fungi, plants, etc., to focus instead on computers. Some of the components of a computer are made of rocks. For the most part, components of a computer are like rocks, in that they exist, to all intents and purposes, statically. When an electrical current is applied by an external agency, the chemical construction of some of those very small pieces of rock changes, and electromagnetic states are manipulated in such a way as to correspond with meanings imposed by an external viewer. Computers are designed to be efficient, in the sense that their operations are performed with the minimum of energy waste, doing only what has to be done, and nothing else. As Turing (1950) notes, their electrical currents form a commonality with the brain, but this is too weak a commonality to suggest that a computer works like a brain, or that it has the properties of a brain. For a clearer counter-example, consider the world-wide telephone network, whose entire business is the transmission of electricial signals: we do not conclude that it is functionally like a brain.

It is sometimes proposed that if a computer large and complex enough were built, then it would be conscious. However, when asked why this would be, proponents of the idea

---

Nagel chose the bat for his example because it is a higher animal with very significantly different sensory capabilities from a human. Thus, people do not find it difficult to attribute consciousness to a bat, but they do generally find it difficult to imagine, for example, how the bat experiences its echo-location system.
have no mechanistic answer. Therefore this solution must be rejected along with other non-scientific proposals and fables of the supernatural.

What matters is that there exists no evidence whatever that any entity exists, made of the same materials and working by the same processes as a computer, that is conscious. On the contrary, the available evidence suggests that only living, biological entities are conscious, except in exotic definitions of the concept (e.g., Tononi, 2004), which are highly contested (e.g., Merker, Williford, and Rudrauf, 2021). Absence of evidence is not evidence of absence; however, since no known non-living example exists of Nagel’s kind of consciousness, a scientist wishing to propose consciousness in non-living circumstances must provide an account of how, or at least where, it arises, in order to be philosophically convincing.

A categorical range of consciousness is expressed in the Indian “scale of sentience” (Merker, 2007):

“This”
“This is so”
“I am affected by this which is so”
“So this is I who am affected by this which is so”

Note that this is a scale, not of magnitude, but of ordered categories of successively superordinate kind. We can place humans in the fourth category. We speculate that dogs, for which evidence of self-awareness is lacking, but which seem capable of feeling how the environment affects them, might be in the third category. Lower molluscs (e.g., mussels) might be in the first, while some cephalopods (e.g., octopuses) are certainly in the third or even the fourth category. Computers and rocks, however, are not in any of these categories, according to any extant evidence.

What it is like to be a conscious entity includes what it is like to experience the environment of that entity: this is the first level of sentience, above. The instances of such experience, *qualia*, are themselves contentious (cf. Dennett, 1991), and they are unverifiable because they are *private*, in the strong philosophical sense: they are not directly communicable. To see this, consider the colour called “blue”. One person may use the word “blue” to signify (part of) the experience of seeing a particular object to another person, and the second person may agree that the label “blue” is appropriate for that object. But it is unknowable whether the two people experience the same thing. For the avoidance of doubt, the question lies not in the external stimulus, because the light reaching the eyes of the two individuals may be measurably the same, but in the internal, private response to the stimulus of the two viewers. To introduce unusual terminology: there is a *feeling of blue*, private to each individual, which is experienced simultaneously, and labelled with a common word.

**Computers and the Feeling of Meaning**

Now compare our two humans with a computer equipped with an RGB raster camera. The camera can measure the light reflected from the same scene viewed by the humans, above, and upload a corresponding matrix to the computer. The computer’s processor can compare these measured numbers with a range, and output the symbol “blue” when a pro-

grammer has deemed this appropriate, or when it has learned the necessary association from data. In spite of the fact that the computer seems to have grasped language, a capacity of only the fourth category of sentient beings, it has merely *measured* blue, lacking any mechanism with which to *feel* it; it has not even reached the first level of sentience. The *anthropomorphic illusion* thus produced is beguiling, and will be important later in the current argument. That illusion permeates popular discourse: we talk metaphorically about “what the computer thinks” and “knows”, when what we really mean is that some numbers have been used to represent some information by a programmer. Humans use metaphor to allow us to discuss things whose detail we do not know; doing so is an important part of communication, since it allows us to learn as we converse, by filling in the gaps as we go along. But it has a down-side: it can lead us to attribute capabilities or properties to the target of the metaphor that are incorrect and misleading.

The association by humans of the word “blue” with the relevant * quale * may also be viewed as defining the meaning of the word, “blue”. One can give more objective measures, based on frequency of light, but this is misleading, since human word usage long predates such possibilities. Thus, the memory of the feeling of blue is the meaning of “blue” for each person who has seen a blue thing, and who uses the word. The word “qualia” is conventionally reserved for sensory experience, such as this. We posit that it is reasonable to extend the concept of sensory feelings, as exemplified here, into more abstract knowledge. Consider, for example, the idea of small, integer numbers. To think of the abstract concept of “two”, a human need not see two things, nor hold up two fingers, nor count up from zero; further, the concept of “two” does not exist independently in the world, but only as a relation between things that exist in it. Nevertheless, humans have a feeling about what “two” means, which may be expressed in arithmetic, but which can still be felt without such expression, exactly as the reader did when considering the “blue” example, above. The contention here is that meaning is a construction of consciousness in context of memory, which begins with qualia building blocks, and which can assemble arbitrarily complex entities from smaller ones. Those entities acquire private meaning by virtue of their very existence, based on that of their components. To make them public, humans must realise them, either literally by building them, or via sequences of word labels that describe them, or via other more precise descriptions, such as mathematical specifications. Such *feeling/meaning*, I propose, is the fundamental stuff of human thinking and therefore of human creative thinking.

Somewhere on the constructive continuum between qualia and philosophical concepts such as “qualia”, lie the *feeling/meanings* that drive biological organisms, some of which are close to, but not the same as, direct sensory experience, and many of which arise from sensing internal states of the body: for example, hunger is the feeling/meaning induced...
by metabolism of glycogen in the liver. More complex are what are generally called aesthetic responses: the feeling/meanings that result from often complex and extended experiences such as music, literature or visual art, and which also arise in humans involved in science, engineering and mathematics, in perhaps less obvious ways. In summary, experience produces feeling/meaning, and knowledge labels feeling/meaning or constructs new feeling/meaning from existing components.

It is for this reason that the research field of computational aesthetics (Hoenig, 2005) concerns itself with the simulation of human aesthetics and not with the development of aesthetics that might be felt by a computer; similar approaches appear in computational creativity (e.g., Norton, Heath, and Ventura, 2013). It is not like anything to be a computer; a computer does not feel, as there is literally no place for qualia. Therefore, a computer cannot have aesthetic experiences, which are, for humans, internally, nothing but feeling/meaning. Below, this will have consequences for the conduct of computational creativity research.

The Nature of Computational Creativity

Hodson (2017, p. 3) suggests that computational creativity, as a research field, has committed a “fundamental misunderstanding” by assuming that creativity is an ex ante phenomenon, rather than an ex post phenomenon. In other words, he writes, the field presupposes that creativity is caused by certain cognitive processes, rather than being something that is observable in context when it happens, no matter how it is produced. This interpretation is generally incorrect. Rather, in computational creativity, many, or even most, researchers seek systems that are imbued with the properties that may afford novel and valued outputs, which may only subsequently—i.e., ex post—be judged creative. In other words, we seek systems with the capacity to be “deemed creative by an unbiased observer” (Colton and Wiggins, 2012). Increasingly more research is devoted to the evaluation of systems qua creative systems (e.g., Ritchie, 2007; Jordanous, 2012; Wiggins et al., 2015; Jordanous, 2018). In other words still, we seek to understand what is necessary in order to achieve creativity (identified ex post), while not expecting to find what would be sufficient (ex ante) for creativity within a given creator. A very clear example of this approach may be found in the Empirical Criteria for Attributing Creativity to a Computer Program (Ritchie, 2007).

The ex post nature of creativity brings with it a danger—the same danger associated with Turing’s Imitation Game, above: is a creative system actually creative, or is it merely appearing to be creative? In an artistic context, this distinction may be viewed as unimportant, or may even be itself the subject of artistic question. In a scientific context, it is problematic, particularly in the context of framing (Pease and Colton, 2011), to which we return below. Here, the gap between what is sufficient for the perception of creativity and what is necessary for creativity will be important.

Returning to the topic of this section: recall the differences between computational creativity and traditional AI. In particular, computational creativity broadens the scope of the intelligent behaviour that it studies beyond that which can be modelled as mere exhaustive search through combinations of solutions to a problem afforded by a representation that is specifically designed for that purpose. Here, the conceptual space of Boden (2004) is paramount, and so is her notion of transformational creativity. This is an operation which changes the very search space, a capacity far beyond traditional artificial intelligence systems. The requirement for a value function which is not restricted to a pre-defined meta-level measure of the pre-established solution space (Algorithm A∗ uses such a restricted heuristic), and for the capacity to exit the conceptual space while still generating valued ideas, modelled as aberration in the Creative Systems Framework (Wiggins, 2006a,b), distinguish this view of computational creativity from traditional AI. Recalling that any algorithm may be written as a search algorithm, the difference might be crudely stated, thus: AI searches for solutions to problems, knowing that the solutions may theoretically exist but not where they are, while computational creativity searches for novel ideas of a particular form, that it values, but does not expect.

Aesthetics in Computational Creativity

Notwithstanding the current, and appropriate, emphasis on evaluation in computational creativity, there are still points where its philosophy becomes moot. It seems generally agreed that internal evaluation in a computational creative system corresponds with the aesthetic response of a human creator to her own work. Thus, that evaluation is a simulation of, or substitute for, the human creator’s feeling/meaning. The same is true in some computational creativity systems. For example, the computational artist, DARCI (Norton, Heath, and Ventura, 2013), derives aesthetic labellings for images by learning from descriptions made by humans.

Colton (2019, §5) develops a roadmap for computational creativity, in seven steps. At level 2, the level of “applicative systems”, a creative system designer must “encode [their] aesthetic preferences into a fitness function”; this, we suggest, is slightly less than DARCI’s learning capability. In level 3, the level of “artistic systems”, a creative system designer must “give the software the ability to invent its own aesthetic fitness functions and use them to filter and rank the images that it generates.” The contrast is clear: at level 3, human aesthetics are out, and machine aesthetics are in. (In parenthesis: the notion of generate-and-test appears to dominate here; whereas it is to be desired that advanced creative systems would not be restricted to that approach, but be rather more deliberate in the construction of their outputs. It seems unlikely, however, that Colton strongly proposes that all “artistic systems” should be limited to generate-and-test.) The problem at level 3 is that there is no discussion of what it means for a machine to have an aesthetic. Given the absence of feeling/meaning in a machine, as argued above, the phrase “machine aesthetic” becomes a contradiction in terms, and therefore meaningless. Perhaps Colton requires his computers to be capable of feeling/meaning? But later he and colleagues say this is not the case (Colton et al., 2020).
Perhaps he intends a sort of arbitrarily generated selector imposing an arbitrary choice, unrelated to feeling/meaningful aesthetic response? But this, therefore, should not be called “aesthetic”. There is no mention of co-creativity (e.g., Kantosalo and Toivonen, 2016) here, and anyway, such collaboration with a human would lead the aesthetic function back towards (if not directly to) something that models human aesthetics, which Colton has rejected.

The only remaining defence is the Intentional Fallacy from literary theory (Wimsatt and Beardsley, 1946): what matters in a work is not what the creator meant, but what the work contains, and what the viewer (or reader, hearer, etc.) experiences. In this context, it does not matter that the “aesthetic fitness function” of level 3 is meaningless (in our specific sense): no feeling/meaning is created, but none is needed. So then the “artistic system” is generating ideas and arbitrarily filtering them.

But what does this mean? An arbitrary, feeling/-meaningless “aesthetic” function selects an arbitrary subset of the items that Colton’s system would generate. Written another way: take the items that the system would generate and choose a random subset according to an arbitrary distribution. This is no different from generating arbitrary items. Thus, Colton’s “artistic system” is doing nothing more than “mere generation” (Ventura, 2016), the most basic form of computational creativity, if one accepts it as computational creativity at all. Therefore, without a meaningful account of the aesthetic function, the distinct levels of Colton’s hierarchy collapse into a single layer.

Alternatively, in Ventura’s terms, Colton’s level 2 is somewhere near “Algorithm 8 (…random generation … and filtering …)” and “Algorithm 9 (…choosing a theme, … acceptable semantics …)”. But because of the collapse of the “aesthetic fitness function”, above, Colton’s level 3 regresses in Ventura’s more precisely elaborated scale, to “Algorithm 4 (Generation …)” — to be explicit: generation without filtering, a step definitively backwards on Ventura’s scale.

We’ve been Framed

Level 4 of Colton’s roadmap is entitled “Persuasive Systems”. Here, the designer has built a “persuasive system that can change your mind through explanations as well as high quality, surprising output.” In this case, the arbitrary, even random, outputs of level 3 have influenced the viewer, and changed her (human) aesthetic sensibilities. A module is added for the software to generate explanations, so that the machine can explain what it did. It cannot explain in terms of feeling/meanings, because it has none, not even ersatz copies of human ones. So either it must explain in terms of syntactic generative steps (for there is nothing else), or it must pretend to have an aesthetic. The latter can easily be achieved by writing in words that relate to human aesthetics:

7The use of random and arbitrary choices in art is, of course, a valid aesthetic decision (Cage, 1973; Revill, 1993). But if that is the case here, then the decision is made not by the machine, but the programmer, contradicting Colton’s premise.

8The ellipses in this paragraph hide parts of Ventura’s definitions that are not specified in Colton’s roadmap.

“I felt…”, “It seemed…”, and so on. Colton’s own system The Painting Fool (eg., Colton, 2012; Colton et al., 2015) and DARCI (Norton, Heath, and Ventura, 2013) both do this kind of text generation. DARCI has explicitly learned its aesthetic and its descriptive vocabulary from humans, and is thus emulating human feeling/meaning. It is less clear, at least to this author, what is the position with The Painting Fool; however, Colton’s text suggests that the utterances are programmed, not learned, which gives them an ersatz feel. The Painting Fool’s website9 contains extensive first-person writing, from the perspective of the system — but this text is written by a human, and not by the system.

In the context of his alternative question, Turing (1950, p. 434) considers the possibility of making a computer look more human to help it win the Imitation Game. He concludes that there would be “little point in trying to make a ‘thinking machine’ more human by dressing it up in … artificial flesh. The form in which we have set the problem reflects this fact in the condition which prevents the interrogator from seeing or touching the other competitors, or hearing their voices.”

So Turing concluded that, in order to answer the question, “Can a machine be mistaken for a human in a sustained written conversational competition with a human?”, one should not frame the machine in a way that assisted its impersonation. Rather, one should be scientifically neutral, and prune away such confounding foliage.

Colton (2019, §3) presents a brilliantly effective explanation, entitled “Computational Authenticity”, of how framing may change the perceived meaning of a poem. The poem, Childbirth, was generated by a computer. But Colton demonstrates how its meaning changes, depending on how it is framed. Its fictional author seems initially female, but then we are told the given name is a pseudonym for a man, and a criminal at that, and finally that neither author really exists. The demonstration is indeed powerful. But then, Colton explains, “We see fairly quickly that it is no longer possible to project feelings, background and experiences onto the author, and the poem has lost some of its value” (Colton, 2019, §3), and we see that he has in fact fallen into the Intentional Fallacy, and not wielded it as defence. Specifically: while the reader may well infer meaning in the poem from their knowledge (correct or otherwise) of its author, it is not what the author thinks that is important in the poem, but what the reader thinks. While projecting onto the maker of an artefact is indeed a pass-time that many humans relish, the resulting conclusions, correct or otherwise, are not part of the poem, but part of the viewer. Thus, they figure in an external evaluation, but not in an internal one, in respect of the creative system that produced the poem. The Romanticist notion that the “value” of a poem lies in projecting back on to what the reader may well infer meaning in the poem from their knowledge (correct or otherwise) of its author, it is not what the author thinks that is important in the poem, but what the reader thinks. While projecting onto the maker of an artefact is indeed a pass-time that many humans relish, the resulting conclusions, correct or otherwise, are not part of the poem, but part of the viewer. Thus, they figure in an external evaluation, but not in an internal one, in respect of the creative system that produced the poem. The Romanticist notion that the “value” of a poem lies in projecting back on to what the author meant or in what they were thinking, was prevalent in the 19th century, but has not been so for more than 50 years (Wimsatt and Beardsley, 1946).

Consider the following thought experiment. The music of Pérotin, a member of the Notre Dame school of composition, around the turn of the 13th Century, is among the earliest surviving attributed music in the West. Almost nothing is
known of this person—even his birthdate and nationality are uncertain. Has Perotin’s music “lost some of its value” because we have no information about him on which to base our own interpretation? Apparently not: his music survives, and is still performed and recorded, after 700 years, a truly exceptional duration in Western culture. Of course, one might argue that the very lack of information contributes value, or at least mystique. But that is the exact opposite of the argument that Colton (2019) is explicitly and unambiguously proposing, so does not refute the Perotin counter-argument.

Digging deeper into Colton’s argument about Childbirth, one sees a pattern. Initially, the poem is presented as the description of life experience by a woman, entering motherhood (deemed, along with apple pie, as “always good”). We are shown what appears to be an expression of feeling about something wonderful, and on which we all vitally depend: we form our own internal explanation of this meaning, as soon as it is offered. Thus, the affective response invoked is not only invoked by the poem, but by the intensification of our emotional connection with motherhood—which is peripheral to the poem. Next, the mother is violently torn away from us and replaced by a repellant person, and we are told that the poem is now about his repellant acts. The poem has not changed, but it is now associated with an explanation that most people will find unpleasant, and that unpleasantness is amplified by contrast with, and loss of, a feeling/meaning of noble and beautiful motherhood. Rhetorical success is clearly afoot, but that success is directly due to Colton, and not at all to his program. Furthermore, the relief that we feel when we learn that the poem was in reality constructed by a machine, and not by a repellant criminal, becomes the central affect, eclipsing the more important fact that really quite a good poem has been produced by a simple computational “cut-up” technique.

The problem here is that the framing of the poem, and the demonstration of its change, while vivid and cleverly executed, is functioning like political “deadcatting”10, (mis)directing our attention away from the important point: the feeling/meanings that really are generated in each individual who reads the poem. That the poem was produced by a cut-up technique is surprising: most such poems will be (much) less good than this one, by chance, so the likelihood is that the outputs of this system were curated, leading us back, again, to Ventura’s pre-creative Algorithm 4. With the dead cat of imaginary authors, Colton directs our attention away from the really interesting possibility: a computational system, capable of representing and reasoning about the syntactic and semantic patterns, and other more abstract images, that are suggested by chance in this poem, and then selecting this poem from other random outputs of the same random process as something of value. That would be Ventura Algorithm 8 or beyond.

Colton and colleagues suggest that creative machines making artefacts about human-centric issues will “naturally be seen as inauthentic” (Colton et al., 2020), in a classic and extreme application of the Intentional Fallacy. To refute this: a further thought experiment has a different man writing Childbirth. This man, aged 60, is a celibate, cloistered, Trappist monk, with no experience of women, nor of the outside world since age 16, and, therefore, no direct experience of childbirth or any of the associated mores. If this man had written the poem, would it be “inauthentic”? Of course not. If we frame the poem with knowledge of the monk, we can see it as a vision of a different life, that he never experienced, or even a religious expression, which is deeply felt and believed, but, in the cold light of day, still not experienced. The construction of such an image in the mind of the monk is no less abstract than the symbols used to infer a corresponding structure in a computer, despite the fact that he probably experiences feeling/meaning as a result of them, while the computer does not.

This thought experiment demonstrates that a poet’s lack of experience of a thing does not render their poem about that thing inauthentic. Indeed, that lack of experience could, for some people, make the poem more remarkable. This applies as much in scientific creativity as in poetic creativity: Einstein did not have the opportunity to experience his physics directly, but imagined abstract things, initially internally, through thought experiments, then externally through mathematics. Only after his death were his ideas empirically validated. Einstein’s abstraction did not make his ideas inauthentic; on the contrary, it made them all the more amazing.

This ersatz notion of “authenticity”, which we suggest is misguided, leads to even more moot philosophy, relating to consciousness, intelligence and humans’ relationships to computers. Authenticity is indeed important to humans, since it is related to trust, and thence comes the current interest in Explainable AI. What makes things authentic, to most humans, is truth—not artistic or absent notions of truth, but simply a thing being what it claims to be. If a man is a known liar, his authenticity is doubted by others, and they do not trust him. If a product does not do what it says on the tin, it does not sell for long. A human artist may construct a persona for herself, and present her art in that context, but if there is a lack of truth in that persona, then the artist is likely to lose the trust of her audience11. This human construct of trust is relevant to the idea of framing in computational creativity, and to artificial intelligence in general.

A Pig in Lipstick

“Framing” is an ambiguous word. It can mean “explicitly placing a created artefact or concept in a particular context”. It can also mean “diverting the suspicion of a crime on to another person”, another kind of misdirection. Misdirection of this kind, if exposed, will backfire on the perpetrator. Colton et al. (2020) propose that computational creativity should adopt the idea of The Machine Condition, by analogy with The Human Condition, with the laudable aim of helping people to relate better to computers. The essential idea is to make computers seem more human-like by attributing their actions to their “life experiences”. Colton et al. (2020) assert

11Lack of detail of such a persona seems more effective than detail, because fans may project what they like on to it. But, then, it is easier to lose faith in a projection than in a reality, when reality intrudes.
that “an entity like a machine does not need to satisfy notions of being alive or conscious to have life experiences worthy of communication through creative expression.” For this not to be an absurd contradiction in terms, we must take “life experience” as a metaphor—for otherwise, how can something that is neither alive nor experiencing consciousness have one? So this is an ersatz notion of life experience, accompanied by no feeling/meaning. Ultimately, the problem is that consciousness is a defining prerequisite of the human condition: what it is like to be a human. There is no machine condition, unless it be a fake one, because it is not like anything to be a machine.

In summary, the idea is to computationally create framing like that of Childbirth (Colton, 2019) as the background to artificial intelligence. The framing would be computationally created, but as artificial as the intelligence that the machine may exhibit. While the facts on which the framing is based may be true, there is no sense in which they or inferences from them are true life experiences, any more than the monk has true life experiences of childbirth, above. If the monk presented these as true life experiences, we would call him a liar, and lose trust in him. The computer has experienced less than the monk—indeed, nothing at all.

Note the important difference between this framing, and the framing of artistic and scientific work in the human world. While human creators may indeed write about their own work, their writing is explicitly presented as such, and not as, for example, authoritative programme notes or exhibition guides. True framing comes from outside the creative system; it arises not from the actions of the creative system itself, but from the social milieu in which the creativity is taking place. If a human artist presented his own writing as the programme note of a critic, he would be a fraud: therefore, framing of this special kind, arising from the creator itself must be explicitly signalled as such, if it is to be honestly presented.

If appropriately and carefully presented, framing can be helpful in understanding. It can also be an entertaining fiction. If left to stand unexplained, or improperly attributed, it is fraudulent and also fundamentally misleading. The danger is only magnified by the beguiling effect of human anthropomorphic illusions about computing machinery. There is little enough understanding of the true nature of computers in the general population, without obfuscating it by pretending, or, worse, faking, humanity and consciousness. Using these terms, even while acknowledging their untruth, is both logically unsound and deceitful at the same time.

Furthermore, a pig in lipstick remains a pig. Eventually, however florid and beautifully gilded the frame, people will see the untruth of the picture it surrounds. Computational Creativity, as a research field, will suffer greatly if its human audience comes to believe it is fraudulent, and more so if the misdirection is deliberate. The frame, even if computationally created, must not obscure even the edges of the truth.

Consequences
What are the consequences of these arguments for Computational Creativity? Some desiderata are now proposed.

1. Creativity is not an ex ante phenomenon, and our research field knows this. We seek what is necessary in general for the perception of creativity, not what is sufficient to a particular case. Let us make this clear.

2. The fact that creativity is an ex post phenomenon, involving the perception of humans, does not entail that we should focus on manipulating that perception. On the contrary, let us investigate the necessary properties for creativity thoroughly, and test them openly and honestly, without obfuscation—even obfuscation that is computationally created.

3. Creativity, intelligence, and consciousness are inextricably linked in humans. While accepting the lack of consciousness in computers, let us study the relationship between creativity and intelligence in the light of that knowledge, with rigorous, philosophically careful arguments, such as those of Turing, seventy years ago.

4. Embracing human-based aesthetics does not prevent a computational system from surprising us or changing our personal aesthetic: indeed, knowing about human-based aesthetics is the first step to reliably challenging their status quo. Since computers are not capable of feeling/meaning, let us not be shy of human-based aesthetics, for we have no alternatives. Let us instead challenge humanity on its own aesthetic terms.

5. Constructing computational aesthetic measures, to be used in a creative context, and managing their interaction with AI techniques that we use for our creative systems, is non-trivial. Let us not be beguiled into framing the shortcomings out of our systems. Let us focus our limited resources, not on illusions, but on solutions.

6. Let us remember that a pig in lipstick remains a pig.

Ultimately, framing based on pretence and philosophical falsehood, no matter how well executed, no matter how well-intentioned, is a beguiling Yellow-Brick Road to an Emerald City of creative systems. At some point, someone will pull back the curtain, the Wizard will be exposed as a fake, and the story will end.

Epilogue and Challenge
On reworking this paper following helpful reviews, I understood, to my surprise, that a key motivation for misleading framing is in fact the ex post nature of creativity itself. If we seek systems that humans will “deem to be creative” (Colton and Wiggins, 2012), then there is always the option of convincing those judges by sleight of hand, supplying what is sufficient for the perception of creativity in a given case, instead of focusing on the necessary computational components and processes that will enable scientific progress towards creative behaviour in computers.

Inappropriate framing of this kind is unlikely to succeed enduringly, because such sufficient properties are, I think, likely to be context-dependent and so unlikely to be general. Therefore the illusion will quickly fade. But this impermanence only renders the clear danger of discovery more present, placing trust in our research field at yet greater risk. It seems, then, that, in order to define our field correctly, we
need a better way of acknowledging its the ex post nature, so as to address both my alternative interpretation of the Imitation Game, and the problem of inappropriate framing in Computational Creativity.

Acknowledgments
The ideas presented here were influenced by insightful conversations with Alan Smaill, at the University of Edinburgh, Graham White, at Queen Mary University of London, and Anna Kantosalo, Simo Linkola and Hannu Toivonen at the University of Helsinki, though I do not claim their agreement with the content of this paper. I am very grateful for these wise and well-informed discussions, and for the remarks of three anonymous reviewers, whose suggestions clearly improved the paper.

References
Colton, S. 2019. From computational creativity to creative ai and back again. InterAlia Magazine.
Countess of Lovelace, A. 1842. Translator’s notes to an article on babbage’s analytical engine. Scientific Memoirs 3:691–731.


Embodiment and Computational Creativity

Christian Guckelsberger1–3†, Anna Kantosalo2†, Santiago Negrete-Yankelevich4‡ and Tapio Takala2†
1 Finnish Center for Artificial Intelligence
2 Department of Computer Science, Aalto University, Espoo, Finland
3 School of Electronic Engineering and Computer Science, Queen Mary University of London, London, UK
4 Department of Information Technology, Metropolitan Autonomous University (Cuajimalpa), Mexico City, Mexico
† Contributed equally and share first authorship
‡ Contributed equally and share second authorship
christian.guckelsberger@aalto.fi, anna.kantosalo@aalto.fi, snegrete@cua.uam.mx, tta@cs.hut.fi

Abstract

We conjecture that creativity and the perception of creativity are, at least to some extent, shaped by embodiment. This makes embodiment highly relevant for computational creativity (CC) research, but existing research is scarce and the use of the concept highly ambiguous. We overcome this situation by means of a systematic review and a prescriptive analysis of publications at the International Conference on Computational Creativity. We adopt and extend an established typology of embodiment to resolve ambiguity through identifying and comparing different usages of the concept. We collect, contextualise and highlight opportunities and challenges in embracing embodiment in CC as a reference for research, and put forward important directions to further the embodied CC research programme.

Introduction

Most researchers agree that creativity and intelligence are closely intertwined (Kaufman and Plucker, 2011). Moreover, it is widely accepted that intelligence is conditioned on embodiment (Brooks, 1991; Clark, 1998); with the words of Pfeifer and Scheier, “intelligence cannot merely exist in the form of an abstract algorithm but requires a physical instantiation, a body” (Pfeifer and Scheier, 2001, p. 649). Without insisting on physicality, we conjecture that creativity, as a form of intelligent cognition, is also shaped by embodiment.

This makes embodiment highly relevant across the entire continuum of computational creativity (CC) research (Pérez y Pérez, 2018; Veale, Cardoso, and Pérez y Pérez, 2019), from engineering artificial systems that can be considered autonomously creative (Colton, 2008; Colton and Wiggins, 2012), to understanding creativity in living beings through computational modelling and simulation (Boden, 2003). Here, some of the most striking questions are if reproducing human-like creativity by computational means is at all possible without also reproducing human embodiment (Guckelsberger, Salge, and Colton, 2017; Valverde-Pérez and Negrete-Yankelevich, 2018), and how changes to a system’s embodiment affect its potential creativity.

Unlike psychologists, whose human subjects share similar embodiments, CC researchers can go beyond investigating the effect of embodiment on creativity per se and explore how people’s perception of creativity (Colton, 2008; Colton, Pease, and Saunders, 2018) as exhibited by an artificial system is affected by their own and the system’s embodiment. If such an effect existed, embodiment would have to be considered an integral factor in the evaluation of CC (Jordanous, 2012) to facilitate fairer comparisons between computational systems and with human creativity. Moreover, using this knowledge, researchers could tune a system’s embodiment to improve the perception of its creativity, and, vice versa, how it perceives the creativity of others.

Given these potential ramifications, it is surprising and alarming that “embodiment” seems to have received little attention in CC research. One potential reason for the apparent void is that the very concept is highly ambiguous (Ziemke, 2003). Theories of embodied cognition – from minimal accounts that “rule out anatomy and bodily movement as important” (Gallagher, 2011), to radical approaches that understand cognition as inextricably bound to bodily processes – conceptualise embodiment differently. Moreover, embodied cognition is closely associated with other, popular extra-cranial and extra-bodily theories of cognition, in particular theories of active, embedded and extended cognition. Due to their proximity and interdependence, they are frequently grouped together into the complex of “4E cognition”.

Crucially though, there appears to be very little awareness of this ambiguity and complexity within CC research. At present, it is perfectly imaginable that two researchers excitingly referred to the “embodiment” of their respective system without noticing that they are talking about entirely different things. We consider this a major problem, given that the various types of embodiment likely have radically different effects on (the perception of) creativity. We argue that the advancement of the field through concerted investigations requires the use of common definitions of embodiment.

The first goal of this paper is to counteract this ambiguity and provide a rich overview of what some authors have already coined “embodied computational creativity (CC)” (Saunders and Gemeinboeck, 2014; Guckelsberger, Salge, and Colton, 2017; Colton, Pease, and Saunders, 2018) research. Based on a systematic review of related work at the International Conference on Computational Creativity (ICCC) as the prime and domain-agnostic venue of CC research, we answer the following research questions:

**RQ1:** What types of embodiment have been embraced in CC research, and how has the usage evolved over time?
RQ2: Why did CC researchers embrace these embodiment types in their work, and what challenges did they face?

RQ3: What does CC research reveal about the relationship of embodiment and (the perception of) creativity?

To counteract ambiguity in the usage of the embodiment concept, we extend and apply a well established typology of embodiment informed by cognitive science to assess the specific types addressed in each relevant contribution. By making transparent which types of embodiment have been embraced, and by highlighting our challenges in assessing them, we want to provide a frame of reference for researchers to adequately and unambiguously address questions of embodiment in their work. Our insights moreover allow us to provide recommendations for an embodied CC research programme – the second goal of this paper.

Types of Embodiment

Much research focuses on distinguishing theories of embodied cognition (e.g. Gallagher, 2011), but comparisons of the underlying and varying conceptualisations of embodiment are rare. To disambiguate different uses of the embodiment concept in CC, we adopt and extend the well-established typology by Ziemke (2003), who distinguishes six types of embodiment informed by research in cognitive science and robotics. We introduce three additions to this typology (one additional type, two additions to existing types) based on more recent insights, and highlight them in italics below.

**Structural coupling**, characterising systems that can perturbate, and, vice versa, be perturbated by their surrounding environment (Varela, Rosch, and Thompson, 1991). Such perturbations facilitate a minimal interaction between the system and environment, in which each has the potential to affect the other’s state (Quick et al., 1999).

**Historical**, characterising systems whose present state is the result of a history of structural couplings, developed through interactions with the environment over time (Varela, Rosch, and Thompson, 1991; Ziemke, 1999).

**Virtual**, characterising simulated systems embedded in and distinguished from a simulated environment. The virtual body can act on the environment and vice versa.

**Physical**, characterising systems with a physical body (Brooks, 1990; Pfeifer and Scheier, 2001) that can interact with the environment by being subjected to and by exercising physical force. Most prominently, robots are physically embodied (Pfeifer, Iida, and Bongard, 2005).

**Organismoid**, characterising virtually or physically embodied systems with the same or a similar shape and sensorimotor equipment as living organisms, e.g. animals. We consider humanoid embodiment as approximations of the human body a subset of organismoid embodiment.

**Organismic**, applying to living and artificial systems capable of organisational closure, i.e. of maintaining their organisation and surrounding boundary against internal and external perturbations by means of self-producing processes (Von Uexküll, 1920; Maturana and Varela, 1987). A prominent, minimal example is the living cell which, in a self-referential process, maintains its organisation, including its membrane, against perturbations from the surrounding environment (Agmon, Gates, and Beer, 2016).

We briefly justify our additions. We have complemented physical with virtual embodiment, because AI researchers have successfully reproduced (super-)human cognitive abilities in virtual agents, embedded in e.g. high fidelity physics simulations (Lillicrap et al., 2015) or, often more coarse, videogame worlds (Mnih et al., 2015). Applying AI techniques to virtual agents in simulated worlds rather than to physically embodied systems allows for scalability, incremental development, and rapid iteration, amongst other advantages (Kiela et al., 2016). As a corollary, we have extended organismoid embodiment to virtually embodied systems, e.g. in the form of believable game characters with a human or animal-like appearance. We have finally extended organismic embodiment to artificial systems. Originally restricted to the biochemical domain, this type required the capacity for autopoiesis, i.e. self-production, to facilitate a radical form of autonomy. Varela overcomes this limitation by introducing the concept of organisational closure as “operational characterization of autonomy in general, living or otherwise” (Varela, 1979). Froese and Ziemke (2009) advocate that organisational closure can be realised by AI systems, and survey existing examples. We can thus consider organismic embodiment in artificial systems, which makes it relevant for CC research. Although organismic embodiment is arguably the least well-established type, we include it for its presence in existing CC theory (Saunders, 2012; Guckelsberger, Salge, and Colton, 2017), its potential future implications for CC, and its central role in related debates, e.g. on agency (Polani, Ikegami, and Biehl, 2016).

In our adaptation of Ziemke’s (2003) typology, we have dropped what e.g. Dautenhahn (1997) and Barsalou et al. (2003) refer to as social embodiment, because it denotes the use of different types of embodiment to facilitate social interaction (Ziemke, 2003), and is thus orthogonal to, and not at the same “atomic” level, as the other types. Metzinger (2014) has proposed a distinction between 1st, 2nd and 3rd order embodiment based on a system’s computational abilities, corresponding to (1) physical, reactive systems without explicit computation, (2) systems that explicitly represent themselves as embodied agents, and (3) systems that can consciously experience some of these body representations. We disregard this typology as (i) it is not derived from existing work on embodied cognition more generally and serve the specific purpose of grounding (artificial) consciousness as one aspect of cognition, and because (ii) it would only warrant little differentiation of existing work; most CC systems presently fall into a gap between the 1st and 2nd type.

Crucially, the extended typology is only loosely hierarchical. Historically as well as physically and virtually embodied systems are all structurally coupled. Physically and virtually embodied systems in turn can, but do not have to be historically embodied. Organismoid systems can be virtually or physically embodied, but we reserve organismic embodiment to physically embodied systems, as autonomy via organisational closure relies on physical forces that pose a real threat to a system’s organisation and boundary.
Review Method

To investigate how embodiment has been discussed in CC research, we performed a systematic literature review on the proceedings of the International Conference on Computational Creativity (ICCC), the prime venue for CC research, between its inception in 2010 and its latest edition in 2020. We expect our findings to be representative, as ICCC gathers a wide audience of CC researchers and practitioners, and welcomes contributions covering any creative domain, creative practice and aspect of creative cognition (Association for Computational Creativity, 2021). We constrained our review to paper candidates that explicitly mention the words “embodiment”, “embodied”, “disembodiment”, “disembodied”, “embody”, or “embodying”.

We acknowledge that our reliance on the explicit usage of the word may overlook a large amount of potentially relevant papers. This particularly concerns work on robotics, which often does not include explicit mentions of the embodiment of the investigated systems. Simply including the term “robotics” in our search however was impractical, as, for fairness, it would have required to also include any other type of system characterised by the remaining embodiment types. Given the inclusiveness of some types, the amount of potentially related papers would likely go beyond what could reasonably be reviewed in depth. Moreover, we believe that our present approach allows us to identify intentional and thus more informative discussions of the relationship between embodiment and (the perception of) creativity.

For our initial candidate paper selection, we divided the past ICCC proceedings into pdf files, each containing one paper, short paper, demo description, or other peer reviewed publication. We used a wildcard search in Adobe Acrobat reader with the phrase “*em-bod*” and the hyphenated version “*embod-*”. In total we found 99 papers mentioning the word “embodiment”, “embodied”, “disembodiment”, “disembodied”, “embody”, or “embodying” explicitly, with a total of 491 matches to the search phrases.

We then reduced the candidate papers for our final, qualitative analysis in a two-stage process. We first excluded papers which mentioned these words only in the References section (7 papers), or as part of a general list of CC related topics (2 papers). We secondly excluded papers that used these keywords in a merely metaphorical way, such as suggesting that a specific algorithm or system “embodies” certain values (51 papers). This left us with a final pool of 40 papers for in-depth analysis. Figure 1 illustrates the overall usage of the term over the years, thus partly answering RQ1.

We assessed the embodiment of any system described in the remaining papers, either introduced there or through reference to other work, based on Ziemke’s (2003) extended typology. Since explicit definitions of embodiment were mostly absent, this usually required us to look at the specific characteristics of the system in question. To answer our research questions, we also gathered notes on:

• Which challenges did we encounter in assessing the embodiment described in the paper? (RQ1)

• Did we identify any embodiment types that were not yet present in our typology? (RQ1)

Figure 1: Absolute annual numbers of ICCC papers using words derived from ‘embodiment’ in a (non-)metaphorical way. On average, 7.1% of papers use the word ‘embodiment’ or its derivatives non-metaphorically, with the lowest proportion (0%) in 2012 and the highest (11.8%) in 2017 and 2020. Overall the term is used throughout the proceedings, but non-metaphorical usage has increased over the years.

• Why have the authors embraced this particular type of embodiment, and what challenges did they face? (RQ2)

• What did the paper express about the relationship between embodiment and creativity, or between embodiment and the perception of creativity? (RQ3)

Each of the remaining 40 papers was first assessed by one of the four researchers participating in the effort. We then discussed the challenges encountered, and cross-checked each paper with another colleague in the team. Any potential disagreements were resolved in dedicated discussions. Another four papers were removed during this step because the explicit mentions to embodiment were cursory. This reduced the final number of papers to 36.

The answers to the questions presented above were analysed by building an affinity diagram, a qualitative data analysis method used to group data into emergent categories (Rogers, Sharp, and Preece, 2011, p. 286). We first built individual affinity diagrams from the answers collected for each question by grouping similar items together and labelling them. We then merged categories between the different questions, yielding one large affinity diagram as a connected view of different aspects of embodiment in CC. We finally identified overarching themes within this structure.

Findings

We report our findings on the state-of-the-art of embodied CC in individual subsections. We first detail the types of embodiment identified in existing work, thus conclusively answering RQ1. We elaborate on our difficulties in this process later in the discussion section, where we go beyond the present findings and make recommendations for future research. Our review uncovered that existing insights on the relationship of embodiment and (the perception of) creativity are often tightly aligned with researchers’ motivations to embrace a certain type of embodiment in their work. In the second part of our findings, we consequently address RQ2 and RQ3 jointly through themes corresponding to opportunities and challenges for embodied CC. We distinguish each
Table 1: Chronological overview of ICCC (2010-2020) papers mentioning the concept of embodiment explicitly and non-metaphorically. The circles ○, ●, and ■ represent increasing degrees to which the described embodiment(s) match our types. Organismoid embodiment entails humanoid embodiment, but we also discriminate humanoid embodiment separately. For papers describing multiple embodied systems, e.g. a robot and a human, individual embodiment types were combined with a logic “or”. Papers introducing a “concrete system” have been marked; other papers consider theory or abstract systems.

**Embodiment Types**

Table 1 provides an overview of our final paper selection, together with our assessment of the described embodiment. We found that structural coupling and physical embodiment are most common, each appearing in almost twenty papers. Historical, virtual and organismoid mentions appear quite equally, each in about ten papers. Organismic embodiment was only identified in four papers; Guckelsberger, Salge, and Colton (2017) discuss it with respect to machines, while the other three instances relate to humans, which are organically embodied by definition. Over the years, although there is considerable variation, the annual number of papers grows from 1.6 on average in the first half to 3.6 in the latter half of the decade, until there is a sudden peak of 10 in 2020. Mentions of different types follow a similar trend, except that structural coupling gives way to the more specific, physical embodiment in recent years. Out of the 36 papers, 23 present a concrete, embodied computational system, seven are at least partly theoretical or appealing to the embodiment of humans. In another seven papers, we were unable to identify one or multiple types of embodiment at
Our analysis highlights virtual and physical embodiment as strongest differentiators of existing work. It moreover identified a chasm between researchers poised to leverage humanoid embodiment, and those rejecting it for the benefits of other variations of organismoid embodiment.

Through our review, we identified one embodiment type that was not directly present in our extension of Ziemke’s (2003) typology: Loesel, Mirowski, and Mathewson (2020) introduce “cyborg embodiment”, which bridges between virtual, organismoid (anthropomorphic) and physical embodiment. It was demonstrated in their theatrical experiment “AI Improv”, where a chatbot provides sentences to be articulated by a human actor, and receives new prompts to react to from a backstage operator who monitors the on-stage dialogue. The two people thus provide a split actuation and sensing interface to the artificial system, allowing for it to be embedded in the physical environment of the stage.

Opportunities of Embodied CC

Our analysis identified nine opportunities of embracing embodiment in CC. We can distinguish two sub-groups, based on how the themes relate to the concept of interaction. All types of embodiment distinguished in our typology assume at least a minimal form of interaction between an agent and its environment in the form of structural coupling. The first four themes concern how this embodiment-induced split provides opportunities for modelling a specific creative domain, outsourcing computation, letting creativity emerge and stimulating it. The remaining five themes operate on a stronger notion of interaction between agents, i.e. where the embodied agent’s environment comprises interaction partners. In this group, embodiment is considered a means to model co-creativity, ground meaning, facilitate more natural interaction with people, support identification and empathy with the computational agent, and increase the CC system’s creative intentionality and autonomy.

Domain Necessity Any type of embodiment presents the opportunity to model creative processes that unfold between an agent and their environment. Some creative domains may necessitate this split more than others in order to comprehensively model the creative processes within. Dance choreography represents a prime example (Augello et al., 2017; Carlson et al., 2016), but embodiment has also been embraced in e.g. music (Schorlemmer et al., 2014) and painting (Schubert and Mombaur, 2013; Singh et al., 2017) to model creative processes that rely on sensorimotor feedback between an agent and their surroundings.

Outsourcing Computation Embodied CC has adopted several premises of embodied AI more generally, notably the use of physical embodiment to outsource computation into the physical world. Saunders, Chee, and Gemeinboeck (2013) note that physical embodiment allows artificial agents “to take advantage of properties of the physical environment that would be difficult or impossible to simulate computationally” (paraphrasing Brooks, 1990; see also Gemeinboeck and Saunders, 2013). They thus relate to one of the most prominently articulated benefits of embodied AI: the use of the world “as its own model” (Brooks, 1991).

Emergent Creativity While the outsourcing of computation seems a mere engineering benefit at first, it has major implications for the creativity that a physically embodied system can potentially exhibit. In particular, it allows to realise the very premise of systems theories of creativity (Csikszentmihalyi, 1988), the emergence of creative behaviour through an agent’s interaction with their environment, including other agents: “embodiment provides opportunities for agents to experience the emergence of effects beyond the computational limits that they must work within” (Saunders, Chee, and Gemeinboeck, 2013). In the art installation Zwischenräume, the robots’ creative agency “is not predetermined but evolves based on what happens in the environment they examine and manipulate” (Gemeinboeck and Saunders, 2013). This emergence benefits from a controller that is not pre-coded but sensitive to an agent’s changing embodiment. Several authors (e.g. Saunders et al., 2010; Saunders, Chee, and Gemeinboeck, 2013; Guckelsberger et al., 2016; Guckelsberger, Salge, and Colton, 2017) highlight the use of computational intrinsic motivation to this end. A system with a suitable controller can leverage its embodiment to expand its behavioural range beyond what can be anticipated by the system designer, realising novelty as core criterion for creativity (Rhodes, 1961).

Stimulation of Creativity As a specific case of emergent creative behaviour, several researchers argue that constraints imposed through embodiment can stimulate creativity. Drawing on Pickering (2005), Saunders et al. highlight that the “world offers opportunities, as well as presenting constraints: human creativity has evolved to exploit the former and overcome the latter, and in doing both, the structure of creative processes emerge” (Saunders et al., 2010). Guckelsberger et al. (2016) argue in an artistic context, drawing on different embodied CC systems and a thought-experiment on the physically embodied robot society Curious Whispers (Saunders et al., 2010), that overcoming embodiment-related constraints in an environment can necessitate and – given a suitable agent controller – yield creativity. Takala (2015) demonstrates this on a simulated robotic arm capable of inventing new and useful movements when encountering obstacles. This also highlights that creative action is possible without creative reasoning, a distinction which is later picked up by Fitzgerald, Goel, and Thomaz (2017). By utilising virtual embodiment, Takala demonstrates that the effect of embodiment constraints on creativity can be investigated without physical embodiment. However, the use of physical embodiment can better alleviate doubts about the emergent behaviour being truly novel, and not engineered a priori into a simulated environment.

Co-Creativity Many of the analysed papers express a focus on stronger forms of interaction between agents. Two particular modes of interaction are given by human-machine and machine-machine co-creativity (Saunders and Bown, 2015; Kantosalo and Toivonen, 2016). This focus can be explained with the observation that embodiment is a prereq-
uisite for co-creativity. Guckelsberger et al. (2016) highlight that “co-creative and social creativity systems are only meaningful if each agent has a different perspective on a shared world, allowing them to complement each other, and for creativity to emerge from their interaction”. The necessary separation of agent and environment is crucially facilitated by any type of embodiment. This allows for the attribution of embodiment to a given system, based on the systemic nature of the system alone. The next three themes represent additional lenses on human-machine co-creativity.

Grounding Meaning Another central premise of more radical theories of embodied cognition, bordering to enactivism (Varela, Thompson, and Rosch, 2017), is that physical embodiment can overcome symbolic representations and ground meaning in sensorimotor interaction (e.g. Dreyfus, 1992). Colton, Pease, and Saunders (2018) stress that this allows re-representing creative domains in action. As an example, they refer to the Marimba playing robot Shimon (Hoffman and Weinberg, 2010) which represents music as choreography of physical gestures. They also emphasise the grounding of machine “life experiences” as an important factor in increasing the perception of authenticity in CC systems. Related, Wicke, Veale and Mildner exploit robot gestures to provide the illusion of grounding computer-generated stories (Wicke and Veale, 2018, 2020; Veale, Wicke, and Mildner, 2019), thus leveraging embodiment to affect the perception of their robot’s creativity.

Natural Interaction Saunders et al. (2010) are first to stress that physical embodiment allows for CC systems to be embedded in rich social and cultural environments. This enables “computational agents to be creative in environments that humans can intuitively understand” (Saunders, Chee, and Gemeinboeck, 2013). Robotic art installations are highlighted as one means to “gain access to shared social spaces with other creative agents, e.g., audience members” (Gemeinboeck and Saunders, 2013). Existing research often stresses that situating CC systems in physical space realises more natural interaction by established means, and can unleash new modes of interaction. This is explained by physical embodiment affording tight feedback loops (Wicke and Veale, 2018) and providing stronger cues to the human interaction partners (Saunders, Chee, and Gemeinboeck, 2013). In some instances, this interaction is constrained to a few individuuals, e.g. when situating a robot on stage to interact with musicians (Savery, Zahray, and Weinberg, 2020) or actors (Loesel, Mirowski, and Mathewson, 2020). In their art installation Zwischenräume, Gemeinboeck and Saunders in contrast open the interaction to a wider audience in an exhibition space, permitting “the development of significantly new modes of interaction” and “engaging a broad audience in the questions raised by models of artificial creative systems” (Gemeinboeck and Saunders, 2013). Within natural interaction spaces, physical embodiment can “improve the relationship between humans and AI, inspiring humans in new creative ways” (Savery, Zahray, and Weinberg, 2020), e.g. in partnering a human musician with Shimon’s reincarnation as rapper.

Identification & Empathy Within the overarching theme of agent interaction, researchers embraced physical, and in particular organism and humanoid embodiment to facilitate and improve communication and, consequently, to afford identification, empathy and affect between human and robot, or within a society of robots. Gemeinboeck and Saunders highlight embodied action in their installation Zwischenräume as a means for communication; it takes the form of robots creating noises with a hammer which members of an exhibition audience and other robots can perceive and react to. They moreover stress from an enactivist perspective that the robots’ actions provide “a window on the agents’ viewpoint” (Gemeinboeck and Saunders, 2013), thus possibly facilitating more introspection. Wicke and Veale refer to work outside CC to emphasise that, “when identification with the [story]teller is the goal, the physical presence of a moving body with a human shape makes all the difference” (Wicke and Veale, 2018), hence stressing the effect of organismoid (humanoid) embodiment. Moreover, they hypothesise that a “listener that can identify with the storyteller is better positioned to empathize with the story that the teller wants to convey, especially when that story is crafted from the life experiences of the listeners themselves”. This reliance on life experience resonates with Colton, Pease and Saundar’s (2018) previously mentioned factors to improve the authenticity of CC systems.

Intentionality & Autonomy While organismic embodiment is rarely addressed in the literature and typically only through human embodiment (Kirsh, 2011; Alexandre, 2020; Loesel, Mirowski, and Mathewson, 2020), Guckelsberger et al. (2016) highlight that organismic embodiment realised in artificial systems might play a central role in future CC research. Their theoretical investigation sets out by considering non-artistic creativity in simple computational systems through the lens of autopoietic enactivism (Maturana and Varela, 1987) as adopted in the theory of enactive AI (Froese and Ziemke, 2009). The latter theory holds that machines with organismic embodiment can, similar to living beings, realise an intrinsic purpose by maintaining the precarious existence induced by this form of embodiment. It moreover claims that this intrinsic purpose can ground intentional agency. Guckelsberger et al. (2016) consider more specifically when organismic embodiment can ground intentional creative agency, realising genuine creative autonomy (Jennings, 2010). They argue that a machine grounds value and novelty in creative activity through the maintenance of their precarious identity, based on acts of self-production and adaptation against entropic forces. The claim that intentional creative agency is contingent on organismic embodiment allows for additional, radical statements. Guckelsberger et al. (2016) argue via Dreyfus (2007) that a CC system might have to accurately reproduce human organismic embodiment to exhibit human-like creativity with intentional agency. Moreover, they introduce the concept of “embodiment distance” to put forward hypotheses on the impact of organismic embodiment on the perception of creativity: “when we evaluate the creativity of non-human systems with intentional agency, we are likely to misjudge value
in their behaviour or artefacts, or hesitate to attribute any value at all, as our embodiment distance is too large” (Guckelsberger et al., 2016). This makes the mimicking of organismoid, and potentially, humanoid embodiment relevant. Colton, Pease, and Saunders (2018) extend the concept of embodiment distance to non-organismically embodied systems and discuss ways to overcome it to foster the perception of creativity and authenticity in CC systems.

Challenges of Embodied CC

Our analysis exposed that many opportunities for embracing embodiment in CC have a flip-side, the impact of which is mediated by the respective embodiment type. The identified three challenges are easily overlooked, as they are often addressed separately from the corresponding opportunities.

Computational & Design Costs While affording a range of opportunities, such as more natural interaction, the stimulation of creativity, grounding, etc. – being embodied in our physical world also puts high demands on [the] hardware, software and system engineering (Saunders et al., 2010) of physically embodied agents. Fitzgerald, Goel, and Thomaz (2017) particularly lament the increased processing costs due to the high dimensionality of robot sensors and actuators. Gemeinboeck and Saunders summarise that “embodying creative agents and embedding them in our everyday or public environment is often messier and more ambiguous than purely computational simulation” (Gemeinboeck and Saunders, 2013). Especially when tempted by the opportunity to outsource computation into our physical environment, these costs must be carefully weighted.

Unpredictability Related, creative behaviour that emerges from the interaction of any embodied agent and their environment, especially if resulting from intrinsic motivation, is often hard to predict (Guckelsberger et al., 2016). This is more relevant in some application domains than others, with many artistic domains affording unique possibilities for playful experimentation. Across domains however, researchers must exercise particular caution when designing for interaction with people. Crucially, virtual embodiment comes with more well-defined interaction interfaces and affords more control in experiments that can be reset and afford stronger introspection.

False Expectations Several authors express hope that organismoid, in particular humanoid embodiment can facilitate stronger identification, empathy and affect (e.g. Wicke and Veale, 2018). Saunders and Gemeinboeck however warn that humanoid robots can cause disappointment as they “elicit human investment based on superficial and often false social cues” (Saunders and Gemeinboeck, 2018). Referencing studies of human-robot interaction (Dautenhahn, 2013), they particularly highlight the risk of shifting human expectations in a robot’s social capabilities based on appearance alone. As a workaround, they suggest focusing research efforts on non-anthropomorphic robots, and on generating embodied natural movement as means of identification, instead of a similar form or sensorimotor equipment.

Discussion of Embodiment Assessment

We faced several challenges in assessing the type of embodiment in the selected papers. We briefly elaborate on embodiment (i) classification challenges, (ii) biases, (iii) “under-” and (iv) “over-“attributions”, and (v) typology limitations. The classification (i) of embodiment was complicated by unspecific descriptions especially in theoretical papers. Moreover, some papers related to several systems at once, or exclusively addressed human rather than machine embodiment. Sometimes the lack of specifics did not allow us to gain insights into the use of embodiment in CC. Schorlemmer et al. (2014) for instance only appeal to embodied cognition in a side-note. Affected contributions are listed in Table 1 without an assessment of their embodiment type.

Biased views of embodiment (ii) were expressed when authors only explicitly recognised embodiment in humans, e.g. in the case of a virtual system affecting the embodiment of a human user, or when discussing the challenges of modelling human movement. An example of this can be seen in Schubert and Mombaur (2013), who attempt to capture the movement dynamics of human painters.

Some authors “under-attributed” (iii) embodiment in that they leveraged a specific embodiment without explicitly referring to its type or properties. We observed this particularly often for organismoid embodiment, e.g. in Saunders et al.’s (2010; 2013) description of the Curious Whispers robot society. Here, the robots’ bug-like, organismoid embodiment is not discussed explicitly, although it may have a specific effect on the perception by a human interaction partner.

Closely related, other authors “over-attributed” (iv) embodiment in that they explicitly appealed to a certain type of embodiment without fully implementing or discussing its required properties. For example de Melo and Gratch (2010) describe an experiment performed on “virtual humans”, but they are used as passive mannequins to shine light on. Given that a simulated environment to perturb and be perturbed is absent, we cannot even attest structural coupling. Again, this was particularly evident for organismoid embodiment.

Limitations of our typology (v) made it challenging to identify specific embodiment types, in particular historical embodiment: several systems presented memory or learning capabilities, but this learning did not necessarily happen during the systems’ lifetime. This launched a discussion amongst the authors on whether e.g. the existence of learning hardware is indicative of historical embodiment. We eventually agreed that historical embodiment is independent of a lifetime criterion or learning, and present if the system’s past structural coupling has an effect on its future coupling, e.g. when a past perception triggers a change in a virtual or physical sensor, e.g. affecting a camera’s angle, thus influencing future perceptions. Organismoid embodiment was also difficult to assess, as the proximity of a system in shape and sensorimotor equipment to living beings is a continuum. Given the focus of many systems on humanoid embodiment, it would have been helpful to consider it a separate type.

Some types of embodiment turned out to be worse at differentiating and distinguishing existing work and insights than others, but this does not necessarily disqualify them from future use. Structural coupling for instance is the most
inclusive type of embodiment and applied to almost all reviewed systems, in particular to any physical object; yet, it proved valuable for sanity-checking whether a simulated system can be considered embodied at all. *Organismic embodiment* in contrast was the most exclusive type. We think it should be included nonetheless, given that existing theoretical work (Guckelsberger, Salge, and Colton, 2017) assigns it a major role in future CC research, e.g. on creative autonomy. As for any other type, the current lack of differentiating power might indicate an under-appreciation in existing work, rather than a weakness of the type itself.

**Directives for Embodied CC Research**

Our analysis supports the hypothesis which motivated this systematic review in the first place: creativity in artificial systems, and how it is perceived, is affected by embodiment. If we assume this hypothesis, then furthering our insights into embodied CC will play a critical role in advancing the goals of CC more generally. However, existing research is highly fragmented and ambiguous, lacks generalising empirical results, and rarely trades-off the opportunities and challenges of a certain type of embodiment. We translate findings from our review on RQ1, RQ3 and RQ2 into three directives as pillars of a future embodied CC research agenda.

**Clarify embodiment:** Our review highlighted that many opportunities and challenges of embodied CC are exclusive to a specific type of embodiment. However, we experienced serious difficulties in assessing the specific type of embodiment embraced in existing work, as documented in the corresponding papers. In order to establish an efficient, unambiguous, verbal and written scientific discourse on embodied CC, we urge researchers to always clarify what specific embodiment they appeal to in a particular theoretical or applied project. To this end, adopting definitions from typologies as presented here may serve as a shortcut, foster comparison, and alleviate the “under-” or “over-attribution” of a certain embodiment. However, any typology can be contested and carries the risk of trading-off opportunities and challenges of a certain type of embodiment. We translate findings from our review on RQ1, RQ3 and RQ2 into three directives as pillars of a future embodied CC research agenda.

**Conduct empirical studies:** Our review moreover uncovered that no existing CC study produced generalising, empirical insights about the effect of embodiment on (i) an artificial system’s creativity, and (ii) how its creativity is perceived by others, including humans. Researchers either make assumptions on this relationship, or draw on existing empirical findings from other fields that may not easily translate to computational systems (e.g. Brown, 2016). We recommend to conduct qualitative and quantitative empirical studies on the impact of a specific embodiment, treated as independent variable, on (the perception of) creativity. Informed by our analysis of which embodiments differentiated existing work most, we recommend that initial empirical studies should investigate *virtual* and *physical*, or *humanoid* and other variations of *organismoid* embodiment, as values of the independent variable.

In evaluating the perception of creativity (ii) as dependent variable, experimenters must eliminate or weight for creative ability (i). Vice versa, in evaluating creative ability (i), they must avoid bias by the perception of creativity (ii), e.g. by employing objective measures of creativity (Ritchie, 2007). When employing subjective measures, researchers should consider concepts introduced in the literature such as the *embodiment gap* (Guckelsberger, Salge, and Colton, 2017; Colton, Pease, and Saunders, 2018) between the system and its evaluator(s) as mediating variable.

**Trade-off opportunities and challenges:** We encourage researchers to eventually make use of these empirical insights for the design of systems that reliably leverage a certain embodiment as a means to an end, e.g. to accurately model creative processes that emerge from sensorimotor interaction in a specific domain. Crucially though, our review showed that most embodiment-related opportunities come with a challenging flip-side. In order to avoid an unfavourable trade-off, we recommend researchers to always inform their choice of a particular embodiment not only by its opportunities, but also by the corresponding challenges.

**Conclusions and Future Work**

Motivated by the potential impact of embodiment on creativity and its perception in artificial systems, we set out to map the present landscape of embodied computational creativity (CC), and offered directions for future research in this area. To this end, we conducted a systematic review of papers presented at the International Conference on Computational Creativity (ICCC) that explicitly discuss embodiment. To counteract ambiguity in the concept’s use, we adopted a well-established embodiment typology, and extended it based on the recent scientific debate. We found that most existing work can be differentiated by its focus on *virtual vs. physical*, and, more fine-granularly, on *humanoid vs. non-humanoid, organismoid embodiment*. Moreover, we showed that each type comes with its unique opportunities and challenges as flip-sides of the same coin. Overall, we identified nine opportunities, e.g. the outsourcing of computation or support for more natural interaction, and three challenges, e.g. unpredictability and the shaping of false expectations, from existing studies. We identified several shortcomings of existing work that likely hinder progress on embodied CC research, most prominently ambiguity in the use of the embodiment concept and a lack of dedicated empirical research. We leverage these insights in our final contribution: three directives to advance embodied CC research.

While we chose the scope of this study to provide a reasonably unbiased, big-picture view of embodied CC, future work should be dedicated to incorporating additional, relevant references. How to meaningfully constrain the scope is an open question, as relevant work can not only be found in be found in CC books, but also related fields such as videogame AI, robotics, design and art. To allow for fairer and more direct, unambiguous comparisons, we moreover suggest to complement the present review methods with in-

---

Proceedings of the 12th International Conference on Computational Creativity (ICCC ’21)  
ISBN: 978-989-54160-3-5  
199
terviews, in which researchers in embodied CC are asked directly to describe the embodiment in their concrete system or theory. We also deem it worthwhile to conduct a separate, deeper investigation of embodiment in co-creative systems and creative system societies, drawing on theories of embodied, embedded, extended and enactive cognition (e.g. Barsalou et al., 2003; Dautenhahn, 1997; De Jaegher and Di Paolo, 2008), and on embodied interaction (e.g. Dourish, 2001) as well as embodied aesthetics research (e.g. Scarinzi, 2015). Together with these extensions, our review and the extended typology could eventually benefit a longitudinal analysis of embodied CC, identifying trends in the attitudes towards using specific embodiment types and their associated opportunities and challenges over time and across venues. We hope that this paper provides the necessary knowledge, inspiration, and guidance to drive future research on embodied computational creativity (CC).

Acknowledgements

We thank our reviewers for their valuable feedback, and Rob Saunders for commenting on the camera-ready paper. We are grateful to Marcus Schuennemann for advising us on social embodiment. CG is funded by the Academy of Finland (AoF) flagship programme “Finnish Center for Artificial Intelligence” (FCAI). AK & TT are funded by the AoF project “Digital Aura” (#311090). SNY is funded by CONACYT; project “ReNACE” (CB A1-S-21700).

References


6. Theory & Practice
Should Machines Evaluate Us? Opportunities and Challenges

Kuakou Bossou and Margareta Ackerman
Computer Science and Engineering Department
Santa Clara University
Santa Clara, CA 95053 USA
{cbosou,mackerman}@scu.edu

Abstract

Generation alone does not make a Computational Creativity system. But, what about machines that only evaluate? When it comes to co-creative systems, humans often take on the primary evaluative role, while machines assist with the generation of creative artifacts. In this paper, we propose flipping the paradigm, envisioning machines that (only) evaluate humans. Challenges and opportunities in this new direction are discussed.

Introduction

Evaluation is a critical aspect of Computational Creativity (CC). In fact, systems that generate without evaluating have been called “mere generation” (Ventura 2016), suggesting that without evaluation, a program should not be considered a CC system. In the co-creative paradigm, the system often takes on the role of the generator, while the human evaluates. In fact, while it can be argued that computers are better at generating, humans retain an advantage in our evaluative capacities (Karimi et al. 2018).1

In contrast to generative systems, here we propose the study of evaluative systems. To differentiate from evaluators in other spaces, we also refer to such systems as “creative evaluators.” We ask a daring new question: What happens when a system only evaluates, and does not generate? What role can such systems play in Computational Creativity? From a co-creative standpoint, we explore an extreme point on the continuum, asking: What if we had a co-creative system that only evaluates?

The concept of a machine evaluating humans has been explored in other spaces, primarily in contexts where a convergent solution is desired. For example, machines have been used to evaluate mortgage applicants (Chen, Guo, and Zhao 2021), grade essays (Santos, Verspoor, and Nerbonne 2012; Ramalingam et al. 2018), judge startup pitches (Hu and Ma 2020), and inform investment decisions (Wu and Gnanasambandam 2017; Bento 2018). In these cases, the problems were formulated as machine learning models through a convergent lens, assuming that there is a single ground truth and the aim is to accurately score or classify.

In contrast to these types of evaluative systems, creative evaluators face a divergent problem, assessing the quality of creative artifacts where there often is not, and typically should not be, a ground truth solution. In this context, the creative evaluator can provide value to a human creative partner by not only evaluating the result, but also by providing helpful feedback. This iterative process can consequently form a meaningful co-creative experience between the human and machine, without the machine ever (directly) engaging in generation.

In this paper, we present our vision for creative evaluators. We contrast creative evaluators against evaluators in other spaces, and discuss the particular challenges of building evaluators for interaction with humans on creative tasks.

Previous work

The Computational Creativity literature stressed the importance of evaluation. Veale and Pérez y Pérez (2020) write that evaluation is a staple of Computational Creativity. Similarly, Ventura (2016) proposes that mere generation systems may not be considered creative. Ventura (2016) writes that “the Computational Creativity community (rightfully) takes a dim view of supposedly creative systems that operate by mere generation”. He further argues that the question of whether a system is beyond mere generation is very closely related to the question of system evaluation, making evaluation part and parcel of creative systems.

Due to the centrality of evaluation in Computational Creativity, CC systems often consist of both a generative and evaluative component (see, for example (Toivonen and Gross 2015), for a discussion of generative and evaluative components of creative systems that utilize machine learning and data mining methods). Note that the emphasis is typically on incorporating evaluation into creative machines that also generate, rather than considering machines who sole purpose is to evaluate.

A number of prominent evaluation methods have gained recognition in the CC community. Ritchie (2007) suggested empirical criteria for evaluating the relative value and novelty of a system output. Colton et al. (2011) propose two formal complementary models (FACE and IDEA) for evaluating creative acts of CC programs. The FACE model...
describes a program creative act whereas the IDEA model embodies notions related to the impact of the creative act. Jordanous (2012) proposed a three-step Standardised Procedure for Evaluating Creative Systems (SPECS). Pérez y Pérez (Pérez y Pérez 2014) proposed a three-layer evaluation model for computer-generated plots. Ventura (2016) presented a spectrum of abstract prototype systems that can be used as benchmarks for evaluating relative creative ability of CC systems. More recently, evaluation of co-creative systems has also been explored (Karimi et al. 2018; Kantosalo and Toivonen 2016).

Evaluation in CC has been studied from three distinct perspectives: A system evaluating its own artifacts (Ackerman and Loker 2017; Pérez y Pérez and Sharples 2001) as in internal process, evaluation of autonomous CC systems (Colton 2012; Pérez y Pérez and Sharples 2001), and evaluation of co-creative systems (Karimi et al. 2018; Kantosalo and Toivonen 2016; Ackerman and Loker 2017). We approach evaluation from a novel perspective, where a machine evaluates artifacts created solely by a human.

Considering machines evaluating humans, there is some relevant work outside of the context of creativity. Those systems include selecting mortgage applicants (Chen, Guo, and Zhao 2021; Thomas, Crook, and Edelman 2017), grading essays (Santos, Verspoor, and Nerbonne 2012; Ramanigam et al. 2018), and startup investment decisions (Wu and Gnanasambandam 2017; Bento 2018). There have even been attempts at making automatic paper reviewing systems (Leng, Yu, and Xiong 2019), although these are in their early stages.

This paper proposes the challenge of introduction evaluation-only systems to Computational Creativity. What role can systems that evaluates, but do not generate, can play in CC? What would it take to create such machines?

Creative Machine Evaluators

In the most common co-creative paradigm, the computer agent’s primary contribution is on the generative side (even if the computer engages in some internal evaluation), while the human takes on the main evaluative responsibilities (even if the human also engages in generation).

For example, the Computational Creativity musical, “Beyond The Fence” was based on an original idea by Simon Colton’s WhatIf machine (Colton et al. 2016). The WhatIf machine engine generates fictional plots using What-if scenarios. Many ideas were generated by the WhatIf machine, allowing the makers of the musical to select one of those ideas, which became the starting point for the musical’s plot.

As another example, ALYSIA (Ackerman and Loker 2017) is a co-creative system that originally focused on assisting users with the creation of original vocal melodies. The machine suggests melody lines for user-provided text, which the user could select, alter, or ask for more options. This process captures role allocation based on the main strengths and weaknesses of human and machine agents.

While co-creative systems where the computer agent’s contribution is largely on the side of generation are common, more balanced models have also been considered. For example, alternating co-creativity (Kantosalo and Toivonen 2016) puts humans and machines on more equal grounds. It is further suggested that the attained results should satisfy both parties.

Instead of placing humans and computers on an equal plane, we seek to study another under-explored interaction, which inverts the traditional co-creative paradigm: What happens if, instead of us evaluating machines, machines evaluate us? In this paper, we posit a new form of co-creativity where the human generates and the machine (only) evaluates.

Imagine, for instance, a machine that assists visual artists. Showing their art to the machine, the creative evaluator will provide meaningful feedback on the art - perhaps commenting on composition, color choices, or even how the art may relate to current events. The feedback may in turn help the artist to improve their work, much like feedback from a human domain expert. Similarly, we can envision a storytelling evaluator, which provides feedback on the story arc, character development, and overall quality of a user-provided story.

In the subsequent subsection, we discuss the landscape of machines evaluators as they exist today, following which we address challenges of building creative machine evaluators.

Taxonomy of Machine Evaluators

In Table 1, we classify machine evaluators in AI into three categories: (1) Convergent Evaluators, (2) Creative Machines that generate and evaluate, and (3) Creative Evaluators (that exclusively evaluate), which is the new category proposed in this paper.

Machine evaluators outside of CC tend to adhere to clear objective functions (“Convergent evaluators”). In contrast to traditional AI, however, CC systems are not inherently convergent. As Ventura (Ventura 2017) says, “There is no such thing as a best song, or best theorem or best design. One cannot maximize a piece of visual art or a recipe or a poem.” As such, evaluators in creative domains must acknowledge the inherently divergent nature of creativity.

The remaining two classes of machine evaluators fit within CC and are consequently divergent in nature. Current CC machines that include evaluative capabilities also engage in generation. While it is not uncommon to find machine agents with creative aims that only generate (“mere generation”), we have not encountered creative machines that engage in evaluation without also engaging in generation. It has been argued that the former (machines that only generate) should not be considered CC machines (Ventura 2016). Our paper expands this dialog to ask what role the latter, machines that evaluate without generating (“Creative Evaluators”), can play in the CC space.

Creative Evaluators Opportunities

Before delving further into the challenges of developing creative evaluators, we discuss the opportunities for CC in this domain.

• Essay competitions may be approached from both a convergent and divergent perspective. Prior work has focused
on correctness-based criteria (Santos, Verspoor, and Nerboune 2012), such as the percentage of errors appearing in the writing. However, there are also opportunities to approach essay evaluation by viewing an essay as a creative artifact, and making a creative evaluator that would both judge and provide feedback on an essay through this broader lens.

- In business, there are applications such as pitch competitions and resume comparison applications (Roy, Chowdhary, and Bhatia 2020). The approaches taken for such evaluators typically fall under the convergent category. However, there are opportunities for building creative business-related human machine evaluators for creative tasks such as company names, logo creation, etc.

- In mortgage and job applications evaluation (Chen, Guo, and Zhao 2021; Thomas, Crook, and Edelman 2017), current machine evaluators are typically ML-based models that are prone to replication or even amplifying the biases found in the data on which they are trained. Creative evaluators may potentially offer an avenue for mitigating this problem if mortgage and job applications evaluation are viewed as a creative task, rather than a convergent one.

- Education offers another arena where creative evaluators can offer value. For instance, educational evaluators can provide personalized feedback to help art, music, or poetry students to improve - taking on a partial role of an educator. Further, the evaluators could provide assessment and grading for educational institutions. We can also envision creative evaluators tackling complex tasks such as curating gallery shows or casting actors.

**Challenges with Creative Evaluators**

In this section, we introduce several considerations and challenges when conceptualizing and developing creative evaluators.

**Divergence**

It is well-established that creativity is composed of quality/value and novelty (Ritchie 2007). Value encourages convergent thinking in seeking quality artifacts while novelty relies on divergent thinking to allow originality. As seen in Table 1, we already have convergent machine evaluators for a variety of applications.

When it comes to creative evaluators, we want these machines to go beyond mere convergent thinking. Primarily, the work that is being evaluated must be a creative task/artifact. More importantly, the machine evaluator must take into account novelty so as to avoid conformity while at the same time not lacking in quality.

Perhaps the primary challenge in the making of creative evaluators is to balance the need for providing concrete, justifiable feedback, while encouraging divergent thinking and pushing the human partner to explore profoundly novel possibilities. Creative evaluators may be modelled after the best educations, who seamlessly combine knowledge transfer with the fostering of divergent thinking, encouraging their students to take big risks into the unknown.

**Explainability**

We propose that having some level of explainability is a core feature of a Creative Evaluator. Recently, Explainable Computational Creativity has been proposed as the study of bidirectional explainable models in the context of computational creativity – where the term explainable is used with a broader sense to cover not only one-shot-style explanations, but also for co-creative interventions that involve dialogue-style communications. (Llano et al. 2020)

In the context of machine evaluators in general and creative evaluators in particular, explainability is important consideration. The European AI Experts (High-Level Expert Group on AI) have been encouraging AI researchers to consider explainability as a core ethical AI principle.

We argue that a creative evaluator must be able to provide feedback to the human involved in the creative process. For example, after the user presents its artifact, the system may provide ideas to help the human to improve the artifact. While full explainability may not always be possible, particularly with ML-based models, higher degrees of transparency are desired.

**Fairness**

How can a Creative Evaluator be a fair judge/evaluator? The Journal of Computational Creativity defines 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" (Journal of Computational Creativity). But, what is an unbiased observer?

Humans are known to exhibit biased behavior. Despite our best efforts, we may carry conscious or unconscious biases that impact how we evaluate (Fiarman 2016). We therefore ask if, in the place of human evaluators, machine evaluators may be less biased?

Machines don’t run the risk of being impulsively subjective. Any degree of subjectivity on the part of a machine is likely to be more consistent. While eliminating bias from automated systems is a real challenge, it is more feasible for machine evaluators to avoid bias based on human agents’ backgrounds (gender, race, etc) when such information is not explicitly provided to the machine evaluator. This form of discrimination may nevertheless happen if careful attention is not paid when utilizing data-driven approaches, algorithms or word embeddings that are being used (Bolukbasi et al. 2016), although there are certainly instances where one’s identity impacts their art in a variety of ways.

Biased evaluation is a serious concern when either human or machine evaluators (or both) are involved. The question is whether in some contexts, fairness may be more feasible to attain through a machine agent than a typical human agent.

Conclusions and Future Work

How can a machine evaluator encourage divergent thinking? How do we practically develop such machines? In this paper, we introduce the concept of creative machines whose only role is to evaluate human-made artifacts. This can be conceptualized as an inversion of the typical co-creative paradigm, where the human generates and the machine evaluates. On the educational front, creative evaluators may act much like teachers, offering ongoing feedback for improvement. In competitive contexts, creative evaluators open up intriguing opportunities for fair evaluation of artistic artifacts.

We hope that this exploration into evaluation-only creative machines will spark discussion in the computational creativity community. If evaluation is indeed central to CC, is it sufficient for a machine to do nothing but evaluate? Turning off generation leads to many interesting applications, may get us closer to true unbiased observers, and gets us to question the very foundation of CC: If evaluation is all we have, will we go out searching for generation?

Acknowledgments

The authors would also like to thank two anonymous reviewers for their helpful comments.

References


Producing creative chess through chess engine selfplay

Wolf De Wulf
Computer Science
Vrije Universiteit Brussel
Computational Creativity Lab
wolf.de.wulf@vub.be

Abstract

This article presents preliminary work on a creative chess engine that can be used to produce creative chess games or sequences. The contribution in this article is the creation of a creative chess engine that is then pitted against itself to form a creative system that outputs chess games. The chess engine is an extension to an existing chess engine that consists of forcing the existing engine to play more creative moves. It is in no way an improvement when compared to existing chess engines, even though it is based on the world’s best: Stockfish. Letting the supposedly creative chess engine play against itself forms a creative system that outputs chess games. Through analyzing these games it might be possible to discover new chess openings or principles.

Introduction

With the advancement of artificial intelligence we can more and more shift from Turing’s question if computers can be seen as machines that can “think” (Turing 1950), to asking ourselves: “Can machines be creative?”, or more specifically in this case: “Can chess engines be creative?” Every chess player has their own style of playing, but so do chess engines. For both, creativity can be very important. Bushinsky (2009) reasoned that there is a correlation between strength of play and creativity. Analyzing this and other observations regarding creativity in chess, inclines Bushinsky to conclude that at this point in time, chess engines are even more creative at playing chess than humans are.

The present creative system is an attempt at formalizing and proving what the majority of the chess community believes to be true: that chess engines can be and often are creative. The creative chess engine is in no way an improvement when compared to the current state-of-the-art. It is rather an extension of the current state-of-the-art – Stockfish – that consists of trying to force the engine to play more creative moves. Note that this means that measures that are able to capture the creativity of a chess move, had to be established. Inspiration was found in the work of Amatzia Avni (1998). Avni analyzed the roots of creativity in human play and reasons on the human creative process when playing chess as well as on general principles that are deemed creative in the game of chess.

Creativity in chess

An important chess concept that one needs to be familiar with to be able to understand creativity in chess, is the concept of openings. In chess it is common for players to memorize sequences of beginning moves. These sequences that have been played for a long time and have been analyzed and concluded to be good are called ‘openings’. In recent years, it has become more and more difficult for humans to find new openings, as a great deal has been tried already. Computer engines, on the other hand, are perfect ‘analyzing machines’. There are multiple cases of computer engines finding new lines in existing openings or analyzing existing openings and concluding that they are sub-optimal when compared to others. As an example take Bushinsky’s Deep Junior discovering a new line in the Sicilian defense that consisted of sacrificing three pawns. Also, in Silver et al. (2017), the authors of the well known AlphaZero engine explain how their engine analyzed the French defense for 2 hours and then stopped playing it completely as the engine concluded that its sequences of moves are sub-optimal.

As mentioned, chess engines have their own style of playing chess. Bushinsky reasons that the engine style is more objective as they do not bring prejudice to their play. They are said to “play the position” and this style of playing is often called the concrete style. Even though chess engines play more objectively and are better at assessing risk, this should not restrain them from playing creatively. An example is the sacrificial move. In the early days of chess engines it was thought to be impossible for them to ever come up with a move that sacrifices a piece with no immediate gain. In 2003, however, Deep Junior sacrifices its bishop in a game in New York against former world champion Garry Kasparov. The strategy of giving up one’s own piece, that was formerly thought of as “god’s gift to man” (Bushinsky 2009), was suddenly implemented by a machine. This occurrence alone should already spark the question if chess engines can in fact play as or more creative than human players.

According to Bushinsky, creativity in chess is not much different from creativity in other domains. He applies De Bono’s (1973) lateral thinking, which consists of: nonconformism, provocation (in the positive sense), flexibility, casting doubt, thinking out of the box and transfer. Some of these more obviously applied to chess than others. Amatzia Avni, who is a psychologist and a chess master, links cre-
ativity in chess to the human’s intelligent process of playing chess. He mentions the following four steps: gathering, synthesis, enlightenment and realization. In his book, called ‘Creative chess’, he explains some general creative elements in the game of chess: nonstandard positioning or functions of chess pieces, the removal of one’s own piece, the breaching of theoretical principles, etc.

From the above, we establish creativity measures that try to capture creativity in chess moves. A chess move always takes into count the complete chess board. This means that if we consider a move on a chess board and we then consider that move on another chess board on which only a single piece has a different position, the two moves are different moves. The creativity that these measures capture is debatable. For the moment, the creative chess engine only uses the four measures below. It should be noted that new measures can easily be added.

**Measure 1 (Unknown move).** A chess move may be creative if it has never been played (in a tournament).

**Measure 2 (Low winrate move).** A chess move may be creative if it has been played (in a tournament) and it has a low win to loss ratio.

**Measure 3 (Sub-optimal capture).** A chess move that captures a piece may be creative if the captured piece is not the most valuable piece that could have been captured.

**Measure 4 (Sacrifice).** A chess move may be creative if it is a sacrifice. A sacrifice is a move that when played, allows the opponent to capture an undefended piece.

The first measure relates to novelty. A move that has never been played before, should be considered creative. The second measure is closely related to openings. When a move has been played before but its win to loss ratio is very low, the chess community and opening theory will evolve to consider it a bad move. However, this phenomenon does not imply that it is a bad move, as it could be that there exists some sequence of moves in which the move can be good. This sequence has just not been found yet. Therefore, playing such a move that is considered to be bad, can be creative. The last two measures are more closely connected to chess theory than to knowledge of played games. They relate to De Bono’s *provocation* and *casting doubt* as they involve playing moves that lose material.

**The creative chess engine**

Figure 1 gives a graphical representation of the structure of the engine. The main job the creative chess engine (De Wulf 2021) has to perform is to calculate three types of scores: optimality scores, creativity scores and hybrid scores. When the creative chess engine plays a game, at each turn it calculates the hybrid scores for all the legal moves and it then plays the move that corresponds to the highest hybrid score.

**Optimality scores** These are the Stockfish scores but normalized by dividing by 100. An exception is when a move \( M \) is part of a sequence \( S \) that forces a checkmate. In this case its non-normalized score is calculated as follows:

\[
\text{score}(M) = 999999 - \text{length}(S) \times 100
\]  

By doing this we force the creative chess engine to play sequences that end in checkmate when they are possible but still allow it to choose between these sequences. For example, when one sequence could be considered more creative than the other, we allow the creative chess engine to choose the more creative one.

**Creativity scores** When a move complies with one of the creativity measures, its creativity score is awarded a weight \( w_{ci} \) specific to that measure. A move’s creativity score is hence the sum of weights \( w_{ci} \), for all the measures it complies with.

**Hybrid scores** The hybrid scores are the creative chess engine’s combination of the optimality and creativity scores. The engine calculates a move’s hybrid score by adding its creativity score, which is itself a summation of weights that correspond to the move’s creativity, and its optimality score, multiplied by a weight \( w_o \).
Modelling the system

There are multiple possibilities for the output of a creative chess system. One could create a system that outputs single chess moves, sequences of moves or even complete games. In this case, the creative chess system is defined to be a system that outputs complete and legal chess games that were played by two of the creative chess engines. By modelling the creative chess system using the Creative Systems Framework (CSF) (Wiggins 2006a; 2006b), it becomes possible to compare strategy games like chess to more artistic and problem solving domains as creative endeavours.

We already defined the conceptual space \( C \) of the system to consist of all the possible legal and complete chess games that are produced by letting two of the creative chess engines play against each other. The universe \( U \) that contains \( C \) could then, for example, contain: partial games, games played by humans, games played by other chess engines, etc. The set of rules \( T \) that allows for the traversal of \( U \) around \( C \), in the case of the creative chess system, consists of rules that define the values of the weights that correspond to each of the four creativity measures that were mentioned in the previous section, as well as the value of the optimality weight. We also need to be able to separate good concepts from bad concepts, or in our case, good chess games from bad chess games. For this, Wiggins introduces a set of rules \( E \), written in \( L \), which may be used to accept or reject concepts in terms of their quality. These rules are discussed in the next section.

In the case of the creative chess system, transformational creativity can be used to transform both \( R \) and \( T \). Transforming the system’s \( R \) can be done in two ways. Firstly, we can transform the ‘selfplay’ constraint to an ‘otherplay’ constraint. To do this, we need to pit the creative chess engine against any other chess engine (e.g. Stockfish or AlphaZero). Since every engine has its own style, we can expect the outputted games to be very different after transforming the system in such a manner. One could reason that after doing this transformation, the conceptual space \( C \) will still be the same. However, there surely are games and sequences of moves that some engines will never play. Therefore, we can safely state that there is a different conceptual space for each version of the creative chess system that uses a different chess engine. These conceptual spaces will almost never be completely indifferent but they will also never be completely equal. Secondly, we can change the rule that requires the outputted games to be complete games. Doing this can again be very interesting, as it could allow us to find chess principles that are not specific to the game itself but can be used in any game. Lastly, transforming the system’s \( T \) can also be done in two ways. Firstly, we can change the creative chess engines’ weights. This method of transformational creativity is natively implemented in the creative chess system and is discussed in the next section. Secondly, creativity measures can be added or removed from the creative chess engines. This method is not implemented in the system but can be seen as possible future work.

Evaluation

Figure 2 gives a graphical representation of the evaluation strategy of the creative chess system. In what follows, the evaluation rules are discussed in detail. Subsequently, their link with the system’s transformational creativity strategy is explained.

Internal evaluation The internal evaluation strategy of the creative chess system should consist of rules that define which outputted games should be accepted as valuable and which should be rejected. To be able to come up with such rules, we first need to define what we find to be valuable in chess games. In this case, inspiration was found in the architecture of the creative chess engine. We have already defined creativity measures for chess moves. On top of that, we also know what moves are supposed to be very optimal moves, through the output of Stockfish. By letting the creative chess engine keep track of which moves it plays, we can, once a game is finished, get the percentage \( p_i \) for each of the move types to the total number of moves that were
played. Also, since two creative chess engines are playing against each other, we get two personal percentage collections per game. Once we have those, writing rules that determine which outputted games are valuable, according to what we defined to be valuable, merely consists of writing rules that verify whether the percentages \( p_i \) are above certain thresholds \( t_i \). Each type of move can have a separate threshold, allowing us to direct the system towards games that consist of a specific composition of moves of our liking. Anyone that is familiar with Ritchie’s criteria (Ritchie 2007) will notice that this strategy of achieving thresholds is very similar to some of the criteria in Ritchie’s model. As mentioned before, in this document the focus lies on the CSF but in future work, due to this resemblance, Ritchie’s criteria could be implemented to further evaluate the creative chess system’s creativity.

Even though this strategy of achieving thresholds as evaluation is a simple one, it can still have its nuances. We can, for example, require that only one of the creative chess engines achieves all of its thresholds while the other engine only needs to achieve some. Note that, if one of the engines achieves all of its thresholds while the other engine only achieves its optimality threshold \( t_o \), the outputted game could still be a valuable one, since one of the engines played in a valuable way, while the other was not utterly useless.

External evaluation Evaluating the outputted chess games externally can be done similarly to the internal strategy. Instead of letting the creative chess engines explicitly count different types of moves, the outputted games can be analysed by a, preferably high-rated, chess player. Multiple approaches can be implemented when doing this. A first one consists of going over each and every move and each time asking the external evaluator for their opinion on the optimality and the creativity of the move. Aggregating the results for each move then results in measures similar to the percentages that are calculated by the creative chess engines. A second, more simple, approach, consists of letting the external evaluator analyse the game by themselves and after that asking them to rate the optimality and the creativity of the game as a whole, on a scale from 1 to 10. In both cases, accepting and rejecting can again be done by comparing the results to specific sets of thresholds.

Transformational creativity Closely related to the evaluation of a creative system is Boden’s transformational creativity (Boden 2004). When a system determines an output to be invaluable, we can let that system evolve into a new system that is more likely to produce output that is valuable. Boden originally only described transformational creativity to be changing the rules in \( R \), which means: changing a system’s conceptual space \( C \). Wiggins (2006a; 2006b) added to that that transformational creativity can also be applied to a system’s \( T \), which means: changing a system’s strategy to explore its conceptual space \( C \). In the case of the creative chess system, transformational creativity has only been implemented to transform \( T \). As mentioned before, this can be done in two ways: by changing the values of the creative chess engines’ weights and by changing the creativity measures for the creative chess engines’ moves. The creative chess system only implements the former.

Each time the system produces a game that is rejected by the internal evaluation strategy, the weights \( w_i \) of every type of move are updated as follows:

- If the corresponding threshold \( t_i \) was achieved, the weight is subtracted a fraction of a given \( \theta \) that is proportional to the surplus of the percentage \( p_i \) to the threshold \( t_i \):
  \[
  w_i \leftarrow w_i - (p_i - t_i) \cdot \theta
  \]

- On the other hand, if the corresponding threshold was not achieved, the weight gets \( \theta \) added to it:
  \[
  w_i \leftarrow w_i + \theta
  \]

Results

Figure 3 depicts two of multiple interesting positions that occurred in a game that was accepted by the creative chess system. In Figure 3a, the black engine sacrifices its knight with its second move. The creative chess engine classified this move as both an unknown move (Measure 1) and a sacrifice (Measure 4). In Figure 3b, the white engine sacrifices a pawn with its fifth move. The creative chess engine classified this move as both a low-winrate move (Measure 2) and a sacrifice.

This game and many others were analyzed and evaluated by a 2000-rated chess player. In their opinion, the games were creatively played but a recurring comment was that the engines tend to play a lot of sacrificial moves. While these moves can be creative, the context in which they are played is very important. There is also a big difference when humans play sacrificial moves compared to when such moves are played by engines. Computers are better at risk-assessment than humans are. We can not calculate a lot of steps a head, which makes playing sacrificial moves very precarious. Important to note is that, during the first few moves of a chess game, there are lots of positions that have been played before and thus will be stored in chess databases. In all of these positions, there is always a small probability of finding a position that has not been reached before or can be reached by playing a move that, most of the time, resulted in a lost game. However, once the game reaches its 8th to 10th move, a combinatorial explosion happens, making every move result in a position that has never been reached before. Of course, in these positions, firstly, low-winrate moves do not exist and secondly, since every move is technically an unknown move, ‘real’ unknown moves do also not exist. However, the fact that the moves that correspond to these two measures have a low occurrence rate does not make them useless. On the contrary, the combination of the two with the two other, theory-based measures, which naturally have a higher occurrence rate, can result in creative games. The external evaluator agreed with this statement and stated themselves that finding other, theory-based measures, to add to the system is possible and could, in combination with the other measures, improve the outputted games.
Conclusions

A creative system that allows for generating creative chess games is presented. In recent years, chess has evolved to a phase where new principles and new openings are very hard to find, when looking for them is done by humans. In that aspect, the creative chess system can be a very handy tool to try to let chess engines do the looking for us. The resulting games can be analyzed by the system internally but for humans to find principles in them, analyzing them ourselves should be valuable. The system has an internal evaluation strategy that allows it to evaluate itself and learn to improve by directing its outputted games towards certain compositions of moves. The system was also externally evaluated, with very interesting results. The outputted games were found to be creative but the sacrificial nature of the chess engines was frowned upon. Sacrificing a lot of pieces can be interesting for chess engines as they are much better at risk-assessment than humans are. However, if humans are to learn new principles from the outputted games, the external evaluator stated that they should contain less sacrifices. In human play, sacrificing pieces is a very scary technique, as we humans can not easily calculate multiple moves ahead. As mentioned in the sections on transformational creativity, reducing this sacrificial nature of the chess engines can be done in two ways: making the corresponding weights smaller, or swapping out the sacrificial measure completely. Both are viable options. With respect to future work on the creative chess system, the latter should be the most interesting. By working together with professional chess players, multiple new creativity measures could be found and implemented in the system.

References

Meta-Evaluating Quantitative Internal Evaluation: a Practical Approach for Developers

Filippo Carnovalini \(^1\), Nicholas Harley\(^2\), Steven T. Homer\(^2\), Antonio Rodà\(^1\), and Geraint A. Wiggins\(^{2,3}\)

\(^1\)University of Padova, via Gradenigo 6, Padova, Italy
\(^2\)Vrije Universiteit Brussel, Pleinlaan 9, 1050 Brussel, Belgium
\(^3\)Queen Mary University of London, Mile End Road, London E1 4NS, UK
filippo.carnovalini@dei.unipd.it

Abstract
Within the field of Computational Creativity, evaluation is one of the most important and more difficult tasks. Sometimes evaluation is part of the creative systems themselves, becoming an internal evaluation. Being a module of a creative system, it is useful to evaluate how effective this internal evaluation is.

In this paper, we propose a procedure for the (meta-)evaluation of internal evaluation modules, that allows for incremental development of both the evaluation module and the creative system, which are considered fully independent of each other. The procedure works by statistically comparing the evaluation of the average output of the generation system with the best results from the same, to see if the evaluation procedure can statistically distinguish the two. We then show how to apply the procedure giving one example evaluating a module we designed to assess structural coherence in generated folk music.

Introduction
Evaluation is one of the most important aspects of Computational Creativity, but this widely used term describes more than one issue that CC practitioners need to face. The most obvious evaluation problem that anyone who ventures into the field soon encounters, is that of evaluating the results of a creative system, and whether the system can be defined creative in its own (Jordanous, 2012). The difficulty of this task called for more precise definitions of what it means to be creative, and, more importantly, what it takes to be creative. This led to the realization that another evaluation problem arises: according to Margaret Boden and other prominent researchers, for a system to be called creative, it needs to be able to explore a conceptual space on its own, and to be able to evaluate what it encounters in non-previous explored areas of said space to find the better results and to guide the exploration (Boden, 2004; Ventura, 2017). This is the problem of what we will call Internal Evaluation – sometimes referred to as self-assessment (Lamb, Brown, and Clarke, 2018), reflection (Pérez y Pérez and Sharples, 2004), or appreciation (Colton, 2008): while generating something is trivial, being able to tell apart good and bad results is what can make a creative system of interest to the community (Agres, Forth, and Wiggins, 2016). This paper addresses both of these problems, by analyzing how an internal evaluation method can be assessed in itself (in what effectively becomes a meta-evaluation: Jordanous, 2014), to tell if the internal evaluation does indeed help in generating better results or if it only becomes another layer of unnecessary complexity.

The approach we propose has the advantage of being relatively simple to implement while developing the internal evaluation module, and is general enough to apply to any kind of generated artefact. Using an existing generation system, the developers must select some results that are considered better than average, and some results that are not cherry-picked at all and can be considered average results from the system. The internal evaluation is not applied to any of these results, meaning that the development of the evaluation module is independent and incremental to the one of the existing generation system, allowing for more flexible development. The artefacts that are thus selected constitute a dataset for the development of the evaluation method. The evaluation can then be applied to the artefacts in the two groups, and a statistical analysis is run on the results. If the two groups show a significant difference in results, the evaluation module can be considered satisfactory and can be implemented within the generation system. Otherwise, further development can be applied to the evaluation method without the need to change the system.

We exemplify this approach by applying a method for the evaluation of musical structural coherence to an algorithm for melody generation developed by other researchers: in doing that we do not evaluate the generation system itself but only our own module for internal evaluation, showing how the two can be completely separate for this meta-evaluation procedure.

Procedure and Examples
In this section we will explain our procedure for meta-evaluation, giving a practical example of how we applied it to a realistic case-scenario, and also giving some indications on how this can be implemented within an incremental development cycle.

Requirements
The proposed procedure, while being general enough to adapt to a wide variety of algorithms and approaches, has
some requirements that need to be addressed before explaining how the procedure works.

The very first requirement is that the system in which the evaluation module needs to be implemented must already be functional. This means that the system must be capable of generating the desired artefact, although the quality of these artefacts might still be in general unsatisfactory (and thus calling for internal evaluation). This also means that the system should not depend in any way on the evaluation module, and the vice versa should be preferred. While this requirement might seem stringent to some, this will also allow for a modular implementation of evaluation modules as well as a facilitated incremental development. In some cases, one might be only interested in the internal evaluation rather than the generation of artefacts. In this case, it can be appropriate to use a system developed by others that generates similar artefacts as those that are under exam. For this paper, that is what we did: instead of implementing a novel music generation algorithm, we used FolkRNN (Sturm et al., 2019), a readily available system for the generation of folk music. This shows the complete independence of the evaluation from the inner workings of the system, that we used as a black box in this work.

The second requirement is the evaluation module itself. Here we will not focus on the development of the evaluation module, which can be viewed in this procedure as another black box. The only condition is that it must accept the kind of artefacts that need to be evaluated as input, and output a quantitative evaluation, i.e., a number. Sometimes the evaluation needs to consider aspects that are “deeper” than the final artefact. In this case, either the evaluation module should be able to analyze them itself to keep the full detachment from the system, or the system must be modified to output those specialized information as well as the final output.

Procedure

**Step Zero: Feature Definition.** The developer should be completely clear about what feature of the artefact the evaluation module should evaluate: saying that it should find the “better” results might be too vague and lead to more biased and less informative results. While this step is more of a premise than an actual step, its importance justifies the inclusion in this section: this step drives the selection of samples for the dataset, as explained below, and not being clear about what is being evaluated will hinder the entire process (Pearce, Meredith, and Wiggins, 2002; Jordanous, 2012). It is also important to choose at this step if the meta-evaluation will be an Holistic Evaluation or a Specialized Evaluation, as explained below.

**Step One: Dataset Creation.** For this procedure to function, it is necessary to create an ad-hoc dataset of output artefacts. This should not be confused with the dataset that was used to train the generation system nor the one that was used to train the evaluation module (if there are such datasets): these are usually human-generated artefacts that are used to define a style or an objective for the system. In this case, the dataset should only be comprised of artefacts that were generated by the system itself. These must be divided into two groups: the “average” results and the “cherry-picked” ones. For the first group, a set of any \( N \) valid results created by the system selected at random (i.e., without any form of selection: ideally, the first \( N \) successive valid products of the system could be used) should be used, with \( N \) big enough to allow statistical analyses to be performed.

The second group must instead be only comprised of results generated by the system that are deemed good. This is an intentionally imprecise term, because it can depend on the objective of the evaluation, and we can distinguish two main types of evaluation, in this sense. An Holistic Evaluation will see how the evaluated feature impacts the results in term of general metrics such as creativity, value, style, novelty. A Specialized Evaluation will instead only evaluate the same feature as the one considered by the internal evaluation module as defined in step zero. These two kinds of evaluation are complementary: the first one is most useful when evaluating the effect of the internal evaluation module on the system in general, while the second is most useful when the internal evaluation module is being evaluated on its own. In both cases, the pieces that will form the second group must be selected by humans with knowledge about the generated artefacts, possibly the author of the system or a group of experts or via controlled questionnaires or crowdsourcing. The method for the selection can depend on the kind of evaluation and the evaluated feature: for example, if an Holistic Evaluation is being performed to find the effect on creativity, the selection could be based on the Consensual Assessment Technique (Amabile, 1983). Vice versa, if a Specialized Evaluation is being performed the author of the system might want to analyze the results on its own to check if they fit his idea of what the evaluation system is supposed to select. Regardless of what method is used, the system authors should be very clear on how the selection was performed both while designing the evaluation and while reporting its results.

Depending on the method used, creating this second group as large as the first group can be extremely time-consuming. Even if it is not possible to include \( N \) elements in the second group, the selection process should start from \( N \) elements, from which the selected results can be extracted. In this way, the numeric disparity between the two groups will reflect the distribution of good results in the average results.

**Step Two: Evaluation** This central step is rather self-explanatory. Each of the pieces selected in the the dataset must be evaluated through the evaluation module, and the results must be collected. As already explained, this procedure requires a full separation of the evaluation from the system, as is clear from the fact that the evaluation happens (long) after the generation is complete. This also allows for some final modifications of the evaluation module at this point, if the process points out some flaws in the software. Yet, the developer should restrain the urge to modify the code to fine-tune it to get better results. While this is feasible and sometimes useful, if the developer chooses to do so step one should be repeated and a new dataset created, otherwise the final evaluation results would be biased.
Step Three: Statistical Analysis  The final step represents the actual assessment of the evaluation module. The results collected on the evaluation dataset represent two different populations, and are expected to be the result of two different probability distributions: if they originated from the same distribution, then it would be the case that there is no difference between the two groups seen from the eye of the evaluation module. Otherwise, if the evaluation module sees a difference between the two groups, there should be a statistically significant difference between the two. To evaluate this, we propose to use the Shapiro-Wilk test to check if the two distributions are not normal. In that case, a non-parametric test like a Mann-Whitney U-test can be used to tell if the two distributions differ. If the Shapiro-Wilk test does not strongly indicate that the distributions are not normal, and there is enough evidence suggesting that the two distributions are indeed normal, the more powerful two-sample unpaired t-test (Welch’s t-test) is to be preferred (Dodge, 2008). These suggestion outlines some of the more common two-sample tests, but other tests may be appropriate depending on the data. All these tests indicate, through the p-value, if the two distributions are statistically different. In general, a p-value lower than 0.05 can be considered a good result, but since we are dealing with artistic features, depending on the applications the developer can be satisfied with a p-value higher than 0.05, if for example there are other features to be considered in the final system or if the developer wants to leave room for “happy accidents” and allow for pieces that do not comply with the stricter definitions of the evaluation metrics, but could be considered good nonetheless and possibly more creative because of how they escape some rules. For this reason, the researchers might aim for less restrictive values (such as p < 0.1) or use the p-value as a continuous metric rather than a boolean one to compare the results of different iterations of a system or different systems. Once again, regardless of the chosen objective, it is important to establish clear goals for this value beforehand to establish if the evaluation can be considered satisfactory or not.

Example

In this section, we review the above procedure by applying it to a specific example to further discuss problems that can arise and other caveats. In doing this, we will meta-evaluate an evaluation method for music generation.

Step Zero: Feature Definition. Our evaluation module tries to evaluate how structurally coherent a piece of generated music is. While it is not important to know the inner workings of the evaluation to apply this procedure, we will briefly discuss how the algorithm works for completeness. This module accepts as input a monophonic music piece with chord annotations, and segments it into fragments that last two measures. The algorithm then recursively simplifies each segment in a manner inspired by Schenkerian Analysis (Simonetta et al., 2018), and then operates pairwise comparisons between the trees constructed from each segment. From all the comparisons operated on a corpus of folk reels from the Nottingham Dataset (Foxley, 2011), some probability distributions that show how those comparisons typically develop are constructed. The new pieces that need to be evaluated are similarly analyzed, constructing trees to describe the new pieces. The evaluation of the piece uses these trees, but instead of constructing probability distributions, we compute the Information Content of the piece’s trees when compared to the learned distributions’ trees. The Mean Information Content, the metric we use in this example, thus serves as an indication of how unexpected (and thus non-typical) the structure of the new piece is when compared to the learned corpus since it compares how common these trees (that represent structure) are with respect to the analyzed corpus. More information about the specifics of this system is available in previous publications (Carnovalini et al., 2021b,a). It is worth noting that the choice of the dataset for the learning of the evaluation module is also to be considered within this point: in this example, it is reasonable to evaluate the output of FolkRNN with the typical structures of traditional Reels, but in other cases having the constraints imposed by a certain corpus could be too limiting. For this example, since we developed the evaluation module but we used FolkRNN, a system developed by others, for generation, we chose to perform a Holistic Evaluation, trying to assess if our evaluation module that evaluates structural coherence is able to impact the general value of the generated output as assessed by FolkRNN’s own community (see next paragraph).

Step One: Dataset Creation. In our case the dataset was created via the FolkRNN web application. For the “average” group, we created ten melodies in the key of C major and 4/4 meter. For the other group, in order to simulate the choice of the system’s creator, we decided not to select the songs ourselves but instead to select those pieces in binary time from the “Tune of the Month” section of the FolkRNN website, that contains pieces deemed most interesting by the developers and users’ community. If we wanted to do a Specialized Evaluation on instead, we should have generated 10 songs and manually select (possibly with the help of expert musicians) the ones that show a good structural coherence.

Step Two: Evaluation Our evaluation system requires the harmony to be annotated on the generated melody, which is not present on FolkRNN generated pieces, so it was manually annotated. This is not ideal, but since we decided to use FolkRNN as a black box we could not modify it to automatically add harmonic annotations as we would have done on a system of our own. Moreover, our system evaluated structural coherence at a fixed length of eight bars, which is the typical length of a period in folk music. FolkRNN outputs pieces that are generally sixteen measures long (not considering repetitions) so each piece gave two samples for the evaluation.

Step Three: Statistical Analysis The “freshly generated” group includes 20 samples, while the “Tune of the Month” has 6 samples, due to the scarcity of songs in the homonymous section of the FolkRNN website. The two distributions of Information Contents seem non normal, according to the Shapiro-Wilk test and the inspection of the Q-Q plot, therefore we used the Mann-Whitney U-test to tell if the two
distributions are statistically different. The p-value resulting from the test is 0.053, a rounding error away from the 95% confidence interval that we aimed for. This is reflected in the boxplot of the two distributions (see Figure 1), that shows how the distributions have different means but also have a rather large overlap. In doing this evaluation, we found out that FolkRNN does a good job at giving a good structure to the pieces it generates, so it is not surprising that our metric is having a relatively hard time distinguishing the selected pieces from the non-selected ones. Figure 1 also highlights one outlier in the non-selected group. It is important at this point to also inspect such samples, as they could give more insights on possible hidden mechanisms and flaws of the evaluation method. In this case, the outlier has a repetition in the middle of the period, repeating the beginning of the two sub-phrases, which would be considered good structure, but fails to give good phrase endings. This tells us that our evaluation module sometimes gives more importance to phrase endings rather than phrase beginnings, an aspect that can be considered in future development.

**Comments and Conclusions**

In this contribution, we introduced a procedure for the meta-evaluation of quantitative evaluation methods meant to be implemented as internal evaluation modules within a computationally creative system. This procedure is based on statistical methods, and while it does require the developers to create an ad-hoc dataset for the evaluation every time a new method must be assessed, it does not necessarily require intervention of external experts, and allows for incremental development of both the creative system and the evaluation method. For all these reasons, we believe it can be useful and practical for researchers and developers at many stages of development. We showed how the procedure functions by applying it to a real case scenario, evaluating a module for the evaluation of structural coherence in folk music, applied to the results of FolkRNN. It is important to mention that this approach did not evaluate FolkRNN directly, nor we mean to make any sort of statement about that system. We only used it as a framework to evaluate our own evaluation module.

We did not discuss the development of the evaluation module nor of the system, since our procedure is meant to apply to a variety of situations, but the question on how to use the results of such an evaluation module within a system might arise. While the answer depends on the specific system, a few possible ways can apply to almost any situation. The most naïve one is to use a threshold on the results of the module to discard any generated results that is too far from the ideal results, or similarly to use the results of the evaluation method to rank the output of the system so that the developer will first inspect those results that are most promising. When more than one internal evaluation feature is implemented, the developers might want to assign weights to every feature to create the final ranking, or might want to use regression algorithms on the evaluated features to find the overall best ranking. Having a variety of evaluated features might also give the possibility to specify settings for the generation/evaluation: in some cases the user might want to be stricter on some rules rather than others.

Regardless on how the evaluation modules are embedded in the final system, it is important to remind that the current proposal is useful for the meta-evaluation of some quantitative metrics used within the system to guide the generation process. On the contrary, this method is not a procedure for the evaluation of the system in itself and for the value or novelty of the final results. To that goal, human assessment done by experts of the artefact’s field is still the method to be preferred (Jordanous, 2012).

**Acknowledgments**

FC is funded by a doctoral grant by University of Padova, and visited Vrije Universiteit Brussel thanks to a mobility grant from Fondazione Ing. Aldo Gini, which made this work possible. NH, ST and GW received funding from the Flemish Government under the “Onderzoeksprogramma Artificial Intelligence (AI) Vlaanderen” programme.

We thank the anonymous reviewers whose comments allowed us to greatly improve this article.

**References**


Sturm, B. L.; Ben-Tal, O.; Monaghan, \\

Software Design Patterns of Computational Creativity: A Systematic Mapping Study

Porter Glines, Isaac Griffith, and Paul M. Bodily
Department of Computer Science
Idaho State University
921 S. 8th Ave, Pocatello, ID 83209 USA
glinport@isu.edu, grifisaa@isu.edu, bodipaul@isu.edu

Abstract
Software design patterns can be helpful in describing the architecture of a system. Our objective is to obtain a broad overview of the current state-of-the-art of software design patterns used in Computational Creative (CC) systems. We conducted a systematic mapping study using manual and snowballing search techniques. Only 7 primary studies are identified in the CC community that explicitly mention the use of design patterns. Within these primary studies, 14 design patterns are mentioned, 12 of which are user-interaction design patterns rather than software design patterns describing the architecture of the system. The small number of primary studies indicates a gap in CC literature regarding the use of software design patterns in CC systems and motivates the need for research to identify software design patterns specific to CC systems.

Introduction
As computationally creative (CC) systems strive to become more creative, they tend to require an increasing number of behaviors. CC systems strive to have behaviors such as self-evaluation, a knowledge-base, and ultimately an understanding of the world — all of which are added to push creative systems further along in the spectrum of creative systems (Ventura 2016; 2017; Glines, Biggs, and Bodily 2020).

Each desired system behavior adds another design challenge when creating a CC system. Each design challenge comes with an opportunity to introduce elements into the system that are not easily maintainable, reusable, or understandable. To avoid common design challenges, system builders can use software design patterns to implement common system behaviors.

Software design patterns, which we will refer to as “design patterns”, are general, reusable solutions to commonly occurring problems in software development. In 1994, a team of researchers referred to as the “Gang of Four” identified the foundational 23 design patterns in their work “Design Patterns: Elements of Reusable Object-Oriented Software” that has since become a standard part of collegiate computer science curriculum (Gamma et al. 1994). Since then, many more design patterns have been identified like model-view-controller, delegation, blackboard, etc. Practitioners use design patterns with the goal of improving maintainability, scalability, reusability, understandability, among other quality attributes (Zhu 2009). However, there is debate as to the effectiveness of design patterns. Some studies have shown that design patterns can negatively affect quality attributes, but conclude that more research is needed (Khomh and Gueheneuc 2008). Though design patterns may negatively affect quality, they have shown to improve maintainability (Zhang and Budgen 2012). Overall, it appears that design patterns should be applied to the problem they solve while considering the consequences they can bring.

The potential benefits of design patterns motivates the desire to identify design patterns for CC systems. Ventura (2017) identifies a general architectural pattern for building a CC system for any arbitrary domain — see Figure 1. In the architectural pattern, a system builder first chooses a domain in which the system operates. Then internal and external representations of artifacts are designed, a knowledge base of the domain is collected, a conceptualization or model is chosen to generate artifacts, an aesthetic is chosen to influence how the system learns, and finally an evaluator for artifacts is designed. Note that the architectural pattern for CC systems describes easily separated components that feed into an overarching architecture: a knowledge base, generator (model), aesthetic, and evaluator. Ventura argues that it is worth spending “significant” time searching for existing implementations for these components rather than building them from scratch. This argument for more reuse in CC systems further motivates the desire for maintainable and easily communicable code underlying these systems.

The goal of this paper is to provide a broad overview to researchers and practitioners of the state-of-the-art in CC systems with regards to the software design patterns used to build them. To provide this broad overview, a systematic mapping study will be performed as described by Peterson et al. (2008; 2015). Our systematic mapping study consolidates the use of software design patterns from 30 papers that present CC systems. This paper aims to provide insights to researchers and practitioners regarding how and when design patterns are used in CC systems to facilitate building systems that are easier to maintain, reuse, and understand.

There are currently no studies providing an overview of design patterns used in CC systems. This paper aims to fill this gap in the literature.
Table 1: Research questions along with their rationale.

<table>
<thead>
<tr>
<th>Research Question</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RQ1</strong></td>
<td>What design patterns are mentioned in CC literature?</td>
</tr>
<tr>
<td><strong>RQ2</strong></td>
<td>Are there software design patterns devised specifically for CC systems?</td>
</tr>
<tr>
<td><strong>RQ3</strong></td>
<td>Of the CC literature that mentions design patterns, what conferences and journals are represented?</td>
</tr>
</tbody>
</table>

Table 2: Inclusion and exclusion criteria for selection of primary studies.

**Inclusion criteria**
- English language articles.
- Peer-reviewed conferences or journals articles.
- Articles published between January 2010 and September 2020.
- Studies that relate to the field of CC.
- Studies that mention a keyword identified as relating to design patterns, whether high level or otherwise.

**Exclusion criteria**
- Studies that do not mention a design pattern of any kind.
- Articles that present frameworks for building CC systems but do not identify or name a design pattern.

Methods

Systematic mapping studies are used to provide a broad but rigorous review of the literature with the goal of revealing gaps in the literature. The systematic mapping study in this paper is conducted as described by Peterson et al. (2008; 2015). Planning of the mapping study is described, including identified research questions and search procedures. The process by which papers are screened is described in the inclusion/exclusion criteria. Then, data extraction procedures are described. Extracted data is kept in a database to be queried for later analysis.

Planning Stage

Identified research questions and their motivations are shown in Table 1. For clarity, we define CC literature as any literature containing concepts related to CC as defined by Colton and Wiggins (2012):

*CC is* the philosophy, science and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviours that unbiased observers would deem to be creative.

Search Strategy

The CC community has yet to establish a formal database of published literature facilitating advanced keyword search. Therefore, a manual search using the search term “design pattern” was performed on the proceedings of the International Conference on Computational Creativity (ICCC). The search term was changed to just “pattern” after an initial search yielded few papers. Once a starting set of primary studies was selected, a forward snowballing search was performed as described in Wohlin’s (2014) guidelines.

Selection of Primary Studies

Identified papers are then systematically marked to be included or excluded from the study. The criteria for including or excluding a paper is shown in Table 2. This criteria is first applied to study titles and abstracts; then to the introduction, conclusion, and methods sections; and finally, full papers are read to evaluate whether the study should be included as a primary study.

Identified Keywords

To address RQ1 and RQ2, we use additional keywords to help detect the mention of design patterns. The 23 design patterns presented by the Gang of Four are included as keywords, e.g., “factory”, “flyweight”, and “mediator”, as well as the three categories “creational...
design pattern”, “structural design pattern”, and “behavioral
design pattern”. Additionally, a keyword is included for the “blackboard” design pattern. Generic keywords “pattern”, “design pattern”, and “software design pattern” are included.

**Data Extraction**

Papers identified as primary studies have the following information extracted from them: *title*, *year published*, *conference or journal*, and *pattern(s) mentioned*. This information is extracted and stored in a spreadsheet\(^1\) for later analysis.

**Results**

There are a total of 458 papers within the eleven ICCC conference proceedings from 2010 to 2020. Within the conference proceedings, a pilot search was performed with the search term: “design pattern”. This preliminary search yielded only four results. Thus the search was expanded by using the more generic term: “pattern”. This new search yielded 195 papers. Of these resulting studies, six satisfied the inclusion/exclusion criteria and were identified as primary studies and as a starting set to perform a snowballing search.

The forward snowballing search yielded an additional 86 new papers from the starting set of six studies. Of the newly found papers, one satisfied the inclusion/exclusion criteria.

The combined search results yield the following seven primary studies: (Compton and Mateas 2015; Concepción, Gervás, and Méndez 2019; Goel 2015; Kreminski et al. 2020; Petrovskaya, Deterding, and Colton 2020; Abdellahi, Maher, and Siddique 2020; Chang and Ackerman 2020).

**Findings Regarding Research Questions**

**RQ1: What design patterns are mentioned in CC literature?** There are 14 design patterns mentioned in the primary studies:

1. Strategy design pattern
2. Instant feedback design pattern
3. Mutant shopping design pattern
4. Chorus line design pattern
5. Simulation and approximating feedback design pattern
6. Entertaining evaluations design pattern
7. No blank canvas design pattern
8. Limiting actions to encourage exploration design pattern
9. Modifying the meaningful design pattern
10. Saving and sharing design pattern
11. Hosted communities design pattern
12. Modding, hacking, teaching design patterns
13. Turn-taking pattern
14. Biologically inspired design patterns

Only one design pattern mentioned is a traditional “Gang of Four” design pattern, namely the strategy pattern. The strategy pattern is used in a story generation system to select and apply heuristics used to weave generated story plot-lines together (Concepción, Gervás, and Méndez 2019). Design patterns 2 to 13 are user-interaction design patterns, e.g., the instant feedback pattern where users observe an artifact, make a change, and observe the result of the change at a glance. We note that user-interaction design patterns inform system architects of how users will interact with the system. However, they do not inform system architects of how objects and classes interact within a codebase, i.e., how a codebase itself is designed, like a software design pattern would. Biologically inspired design patterns describe generic patterns by which biology is used as inspiration to solve a problem. Like user-interaction design patterns, biologically inspired design patterns do not inform system architects of how objects and classes interact.

**RQ2: Are there software design patterns devised specifically for CC systems?** The 11 user-interaction design patterns identified by Compton and Mateas (2015) are presented in the context of CC but are not specific to CC. Out of the seven primary studies, there are no identified design patterns that are specific to CC.

**RQ3: Of the CC literature that mentions design patterns, what conferences and journals are represented?** The one primary study found during the snowballing search came from the 2020 AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE). The remaining six primary studies come from various years of the ICCC. As shown in Figure 2, five out of the seven primary studies were published within the last two years (2019 and 2020).

---

\(^1\)Link to data: https://tinyurl.com/y4w7najp
Discussion

With only seven primary studies, the results indicate that design patterns are not often mentioned in CC studies. In other words, the results indicate a gap in CC literature in regards to the use of design patterns in CC systems.

Perhaps unsurprisingly, the design patterns most mentioned in CC literature are user-interaction design patterns. These patterns are particularly relevant to the CC community in helping design co-creative systems. While user-interaction design patterns do not describe how to design the system codebase, like a more traditional “Gang of Four” design pattern would, they do provide guidance for building the overarching system architecture. Only three papers (Chang and Ackerman 2020; Abdellahi, Maher, and Siddique 2020; Kremsinski et al. 2020) present systems where a user-interaction design pattern is used, indicating that many papers presenting co-creative systems do not disclose the use of a user-interaction design pattern.

The lack of results could indicate that the CC community is unaware that they are using design patterns. In this case, CC researchers would be less likely to mention them. Ultimately, our results indicate that we cannot gain a true understanding of the pervasiveness of design patterns in CC without reviewing the actual CC systems. This suggests the need for an empirical study on CC systems as future work. An additional avenue of research would be the evaluation of Ventura’s architectural pattern for building a CC system — identifying common approaches (or patterns) for implementing each of the components. The suggested future work would achieve an understanding of the patterns used in building CC systems and furthers the idea that Ventura identified an architectural pattern.

We see design patterns in the CC community as an opportunity for pedagogical benefit — a way to bring in new CC community members and programmers by facilitating systems that are easier to reuse and understand.

Conclusion

This systematic mapping study aimed to provide a broad overview of the use of design patterns in papers presenting CC systems. However, the results show that there is a lack of discussion in CC literature regarding design patterns. This motivates future work to identify design patterns in CC systems to gain a full understanding of how design patterns are used in building CC systems. The results also motivates the need to open a conversation on design patterns in the CC community to create CC systems that are easier to maintain, reuse, and understand, facilitating more collaborative research.

References


Chang, J., and Ackerman, M. 2020. A climate change educational creator. In 11th International Conference on Computational Creativity.


Gamma, E.; Helm, R.; Johnson, R.; and Vlissides, J. M. 1994. Design Patterns: Elements of Reusable Object-Oriented Software.


Deciphering The Cookie Monster: A case study in impossible combinations

Maria M. Hedblom
Institute of Artificial Intelligence
University of Bremen, Germany
hedblom@uni-bremen.de

Guendalina Righetti
KRDB, Computer Science Faculty
Free University of Bozen-Bolzano, Italy
ruendalina.Righetti@stud-inf.unibz.it

Oliver Kutz
KRDB, Computer Science Faculty
Free University of Bozen-Bolzano, Italy
okutz@unibz.it

Abstract

In conceptual blending, the transfer of properties from the input spaces relies on a shared semantic base. At the same time, interesting blends are supposed to resolve deep semantic clashes where many concept combinations correspond to impossible blends, i.e. blends whose input spaces lack any obvious similarities. Instead of a shared structure, the blends are based on bidirectional affordance structures. While humans can easily map this information, computational systems for creative constructions require an understanding of how these features relate to one another. In this paper, we discuss this problem from the perspective of linguistics and computational blending and propose a method combining theory weakening and semantic prioritisation. To demonstrate the problem space, we look at the Sesame Street character 'The Cookie Monster' and formalise the blending process using description logic.

Introduction

Conceptual blending (CB) has been proposed as a model of combinational creativity (Boden, 1998; Fauconnier and Turner, 2008). Based on the principle of analogical transfer, where information of one domain is transferred onto another based on their shared structure, CB suggests that creativity arises as a conceptual merge of two input spaces. Cognitively speaking, blending is a dynamic process guided by a series of optimality principles that repeatedly update the interpretation of both the shared structure and the blended concept (Fauconnier and Turner, 2008). In computational approaches to CB, a static knowledge representation is required, limiting the possibilities to model the dynamics of emergent processes in cognition (e.g. Kutz et al. (2014); Pereira and Cardoso (2002)). This becomes a problem when concepts from (very) different ontological branches are merged, a phenomenon called impossible blends (Turner, 1996). While humans can through mental elaboration principles find shared structure and connections between concepts that have little to nothing in common, computers are forced to rely on the information they are presented with and innovative methods to identify potentially shared structure are required in order to advance computational CB.

In comparison to computational blending, the creative process of noun-noun (NN) compound word constructions demonstrate how humans easily can blend words from different ontological branches based on other criteria than directly shared structure. Concepts like coffee cup and face-palm do not result from some manipulation of the intersection of their respective input spaces. Instead, a coffee cup is a cup particularly designated to contain coffee based on the inverse role and bidirectional affordances of how a cup is a container for liquids and how liquids need a container. For a face-palm, there exists a shared ontological structure, namely body-part. However, a face-palm is not a body-part, but rather slang for the emotional reaction captured in the embodied action of covering your eyes with your hand. Computational blending systems that rely on mapping the intersection of two input spaces would not be able to reach these interpretations as they involve an understanding of embodied experiences that is not shared between the input spaces.

In order to formally deal with impossible blends, they need to be treated with respect to the semantic components within the respective input spaces and how they could relate to one another. As in the examples above, some important semantic components are object affordances (Gibson, 1977) and semantic components describing perceptive and embodied experiences, such as those found in image schemas (Johnson, 1987).

To learn how to better deal with impossible blends in computational blending, we present a top-down analysis of the creative process that takes place when conceptualising the impossible blend found in the Sesame Street character The Cookie Monster. Our method utilises ontological weakening of the input spaces to identify shared semantic structure with an emphasis on identifying transitive and inverse roles of affordances and by performing property interpretation in the form of semantic prioritisation. In a miniature setting, we formalise the spaces using Description Logic (DL).

All that combines is not Blending

Compound words are lexical compositions in which one domain inherits properties from another domain by merging two words, e.g. blackbird and coffee cup. In English, two parts can be distinguished: the Head, denoting the class, and the Modifier, restricting the meaning of the word. For instance, a compound concept such as Cookie Monster would be differently interpreted than a Monster Cookie:
in the first case the word Monster plays the role of the Head, modified by the word Cookie (possibly a Cookie-eating Monster, see below). In the second one, it is the other way around, and the concept is more likely to be interpreted as something like a Cookie which is monstrous in some respects.

According to Wisniewski (1997), NN combinations can be interpreted in three ways: 1) The first is the relation-linking interpretation, where some kind of relation between the components is highlighted (the Cookie Monster is a monster that eats cookies). 2) The second is the property interpretation, where one or more properties of the Modifier noun apply to the Head concept (the Cookie Monster is a monster that is as sweet as cookies). 3) The third is called hybridisation, which is a "combination of the two constituents [...] or a conjunction of the constituents" (Wisniewski, 1997, p.169). The result of the combination corresponds essentially to a 'mash-up' or 'blend' of both components (the Cookie Monster would then be both a cookie and a monster).

However, even with this differentiation, the inheritance relationship from the input spaces is not always straightforward. Consider the difference between the NN compound words snowman and the ice-cream man. Ontologically, the input spaces snow and ice-cream share several properties such as being cold and fluffy, yet the compound words are ontologically distinct based on essential properties and what they are used for. In the case of a snowman, the Modifier’s properties are transferred in its entirety as the compound refers to a man made out of snow. Hence, snowman corresponds to a hybridisation of the two concepts. In the ice-cream man, the result has little to do with any properties of ice-cream. Instead, the ice-cream man blend calls for a relation-linking interpretation on weakened input spaces based on functionality. Here, man: ability to bring is treated in relation to the ice-cream space, essentially making the blend a man who brings ice-cream.

In comparison to the linguistic research on compound words, CB is the emergent process that finds this intersection during (primarily) hybridisation. The blend inherits properties from both input spaces and through emergent properties and optimality principles, the blends are ensured to make sense from a cognitive perspective (Fauconnier and Turner, 1998; Pereira and Cardoso, 2003).

Turning such cognitive processes into ‘artificially intelligent’ identification of shared structure and projection of relevant information is a non-trivial problem. One important feature is that (most often) the most salient and semantically rich features should be inherited by the blend. Arguably, the essence of objects are tightly connected to the affordances they offer (Gibson, 1977), especially in terms of their functional, spatiotemporal behaviours.

Affordances have an interesting feature. They are essentially bi-directional dispositions (Beßler et al., 2020) with transitive or inverse roles of the participants (e.g., inverse roles in DL (Horrocks and Sattler, 1999)). For instance, Food has properties that offer the affordance ToBeEaten as its most essential property is to be edible. Simultaneously, a LivingCreature has the behaviour CanEat as it is essential it should eat, else it is not alive (for long). These kinds of essential properties are of crucial importance when performing CB and interpreting compound words and should, therefore, be incorporated in the formal blending process.

Two important suggestions for improving semantics for computational CB have been introduced (e.g., see (Eppe et al., 2018; Hedblom, 2020)). The first is theory weakening, in which the input spaces are ontologically generalised into spaces of less detail, or higher-order components, to better identify potentially shared structure. Axiomatised theory weakening has been applied in logical approaches to analogy and CB (e.g., Gentner (1983); Schmidt et al. (2014)). However, they lack semantic selection. The second suggestion is semantic prioritisation that promotes that the most important attributes and properties should be transferred into the blend. An example is the property interpretation found in the houseboat blend: despite being a house to live in, it also moves on water as the most salient image-schematic affordances of the Modifier, boat, is projected into the blend. In the next section, we demonstrate how these two methods are involved in deconstructing the Cookie Monster blend.

The Complexity of the Cookie Monster

Actually named Sidney Monster, The Cookie Monster is a blue hand-puppet from The Muppets famous for his obsession with eating cookies. The conceptual complexity that emerges when looking at the Cookie Monster as a compound blend is the following: While the character is a Monster by Muppet-classification, the most appropriate interpretation is that the epithet monster is there to describe an unnatural relationship to cookies. Compare it with calling a (non-monster) friend a “cookie monster” if s/he eats a lot of cookies.

From a formal blending perspective, two interesting things happen in this blend:

1) The blended space is not an intersection between the input spaces Monster and Cookie. Instead, it corresponds to a relation-linking interpretation, based on a conceptual mapping between the inverse roles of: edible and canEat.

2) The second thing that happens is that the blend is not exclusively a cookie-eating monster. Cookie Monster is a sweet character that simply eats cookies in a monstrous way (over-consumption, guzzling, etc.). This is a form of property interpretation where the sweetness of the cookies are transferred onto the blend and, even more interestingly, the unnaturalness of monsters are transposed onto the cookie-eating property.

As cookies and monsters have nothing in common in their conceptual spaces, they need to be generalised to the point in which a connection can be made. For this, we use the

---

1 Arguably it would be possible to claim that a snowman is just snow that inherited the shape of a man. However, we pertain that the snowman is inadvertently also given individual identity through anthropomorphism.

2 Hedblom (2020) calls this image schema prioritisation and focus on spatiotemporal relationships. However, the idea can be transposed onto any conceptual components of semantic importance.
ontological branches to identify the first bi-directional affordance relationships and ‘step up’ in the object classes.

In Figure 1, the blending diagram for Cookie Monster is presented with respect to both theory weakening and semantic prioritisation based on property interpretation, explained in more detail below.

**Initiation of Computing the Impossible**

**Step 1: Relation Linking through Weakening.** A computational approach looking for a relation-linking interpretation needs to identify a relation that holds between the two input spaces. In our top-down example, and following a logic-based representation, this corresponds to looking for a role $R$ holding between the instances of the classes in the input spaces. If such a role is identified, then the task would be easily solved. For instance, for the input spaces ‘Human’ and ‘Monster’ one trivial relation-linking interpretation would be $R$:scare as monsters are commonly perceived as dangerous to humans\(^1\). However, for the input spaces Monster and Cookie, ontologically represented in Figure 1, no such obvious relation exists\(^2\).

Addressing this, one (or both) input spaces need to be generalised until a shared role is found. This corresponds to a form of theory weakening, where the weakening allows to ‘step up’ in the ontological branches by exploiting the subsumption relations holding between the concepts in the ontologies. One possibility to formally capture this is to utilise a generalisation operator as described in (Confalonieri et al., 2020) and as exploited in (Confalonieri and Kutz, 2020). In short, a generalisation operator with respect to an ontology is a function $\gamma_O$ that takes a concept $C$ and returns the set $\gamma_O(C)$ of the super-concepts of $C$\(^5\). Intuitively, a concept $D$ is a generalised super-concept of the concept $C$ with respect to an ontology $O$ if in every model of the ontology all instances of $C$ are also instances of $D$.

In our example, the ontological assumption presented in Figure 1 claims that LivingBeing is a super-concept of the concept Monster. Applying theory weakening, the input space Monster is then generalised into LivingBeing, and a relation holding between LivingBeing and Cookie is sought. If a relation is found, it is returned. Otherwise, as in this case, also the other input space needs to be generalised from Cookie into Food. Here, the role eats holds between the instances of LivingBeing and Food and constitutes the shared structure that belongs to the generic space.

Following blending heuristics, the information in the generic space constitutes the foundation for the blend by adding the specific information from the input spaces, generating the blend CookieMonster $\equiv$ eats.Cookie. However, as CookieMonster $\equiv$ Monster is also a correct interpretation, and following the transfer of information between Head and Modifier, the blend will also be defined by the axioms describing the Head ontology of Monster. Yet, one more complexity arises due to the nature of this impossible blend. This leads us to step 2.

**Step 2: Semantic Property Prioritisation.** Semantic prioritisation is a form of formal property interpretation. It suggests that the most salient features should be identified in the input spaces and inherited into the blend. One such interesting semantic transfer is the mapping of the abnormality of Monster into the role identified in the previous step, namely the eats function. This role is enhanced by an abnormal relationship, i.e. to define a role inclusion such as abnormallyEats $\equiv$ eats (Horrocks and Sattler, 1999).

Finally, as with all blending, the Modifier concept is there to alter the nature of the Head concept. For Cookie, the most salient feature, i.e. that distinguishing it from the other foods, is $\exists$hasProperty.Sweetness and its conceptual space extends more than just sugary foods—but also sweet and desirable characteristics. In contrast, monsters are scary and its conceptual space is directly inconsistent with that of sweetness. Directly transferring the salient features of both input spaces, therefore, may create a logical impossibility. Working top-down we already know that Cookie Monster is a charming fellow, hence priority is given to the Modifier. In unknown combinations or situations with multiple salient features, prioritisation strategies need to be applied to identify the most appropriate mapping.

Identifying the salient features to be transferred is one of the biggest challenges for future work.

**Discussion and future work**

Computational blending has become one of the most widely used methods for simulating computational creativity, yet human ability still far exceeds the current state of the art of computational systems. To contribute to this research agenda, we took a brief look into impossible blends by using the Cookie Monster

---

\(^1\)Based on the assumption that monsters are inherently scary (Neuhaus et al., 2014).

\(^2\)In the context of a formal ontology Monster $\equiv$ LivingBeing and LivingBeing $\equiv \exists$ eats.Food implies Monster $\equiv \exists$ eats.Food. Ideally, theory weakening could be set to exploit logical inference directly.

\(^3\)Conversely, also a specialisation operator can be defined.
as a case study of a compound of two ontologically distinct branches. Building on linguistic research on noun-noun compound words, one of the paper’s main contributions to formal CB is showing how theory weakening could be employed to identify relation-linking interpretations, as well as utilising semantic prioritisation of salient features to more accurately deal with property projection.

The ideas follow the large body of work aiming to improve computational blending (e.g. Eppe et al. (2018); Neuhaus et al. (2014); Veale, Seco, and Hayes (2004)). The distinction between Head and Modifier is also reminiscent of asymmetric amalgams as described in (Besold, Kühnberger, and Plaza, 2017), with the difference that our approach does not identify a traditional generic space and uses different computational techniques as well.

Many challenges remain in order to utilise these ideas to deal with impossible blends in computational CB. To address these, future work includes incorporating the work on generalisation operators together with a system for semantic prioritisation. More precisely, by building on previous research and on empirical results regarding concept salience in compound words (Devereux and Costello, 2012), we plan to combine the formal work on inverse and transitive roles (Horrocks and Sattler, 1999) together with an ontological repository of affordances and other semantic components.

Acknowledgements

Special thanks to Prof. Mark Turner for providing valuable input on conceptual blending during the completion of this research.

The research reported in this paper has been supported by the German Research Foundation DFG, as part of Collaborative Research Center (Sonderforschungsbereich) 1320 “EASE - Everyday Activity Science and Engineering”, University of Bremen (http://www.ease-crc.org/).

References


7. Tools & Techniques
Active Divergence with Generative Deep Learning - A Survey and Taxonomy

Terence Broad\textsuperscript{1,2}, Sebastian Berns\textsuperscript{3}, Simon Colton\textsuperscript{3,4} and Mick Grierson\textsuperscript{2}

\textsuperscript{1}Department of Computing, Goldsmiths, University of London, UK
\textsuperscript{2}Creative Computing Institute, University of The Arts London, UK
\textsuperscript{3}School of Electronic Engineering and Computer Science, Queen Mary University of London, UK
\textsuperscript{4}SensiLab, Faculty of IT, Monash University, Melbourne, Australia

Abstract

Generative deep learning systems offer powerful tools for artefact generation, given their ability to model distributions of data and generate high-fidelity results. In the context of computational creativity, however, a major shortcoming is that they are unable to explicitly diverge from the training data in creative ways and are limited to fitting the target data distribution. To address these limitations, there have been a growing number of approaches for optimising, hacking and rewriting these models in order to actively diverge from the training data. We present a taxonomy and comprehensive survey of the state of the art of \textit{active divergence} techniques, highlighting the potential for computational creativity researchers to advance these methods and use deep generative models in truly creative systems.

Introduction

Generative deep learning methods, and in particular deep generative models, have become very powerful at producing high quality artefacts and have garnered a huge amount of interest in machine learning, computer graphics and audio signal processing communities. In addition, because they are capable of producing artefacts of high cultural value, they are also of interest to artists and for the development of creativity support tools.

One of the main goals of researchers in computational creativity and by artists and others using generative deep learning systems, is to find ways to get generative models to produce novel outcomes that diverge from the training data. In some respects, attempting to create a generative model that does not model the training data is an oxymoron, as by definition a generative \textit{model} must model some existing data distribution. However, generative neural networks are powerful tools with the unique capability of learning to render entire distributions of complex high dimensional data with ever-increasing fidelity. It is no wonder then, that there have been a large number of approaches developed in order to actively diverge from the training data, or any existing data distribution.

The term \textit{active divergence} (Berns and Colton, 2020) describes methods for utilising generative deep learning in ways that do not simply reproduce the training data. Methods for this have been developed within the field of computational creativity, but also a goal commonly shared by neighbouring communities, such as those building creativity support tools and artists, researchers and other practitioners publishing and sharing results under the ‘CreativeAI’ banner (Cook and Colton, 2018). This paper offers a comprehensive survey and taxonomy of the state of the art with respect to methods developed across these fields.

Additionally, this paper outlines some of the possible applications, and outlines key opportunities for computational creativity research to advance active divergence methods beyond tricks and hacks, towards more automated and autonomous creative systems. Many of the research directions presented are still very nascent and a lot of work is still to be done in regards to evaluating and benchmarking these methods. Better ways of measuring and evaluating these techniques will go a long way to advancing understanding and allowing more creative responsibility to be handed over to the systems. The comparative account of the methods, use-cases and future research directions for active divergence is offered as a resource to inform future research in generative deep learning tools and systems that take creative leaps beyond reproducing the training data.

Technical Overview

While not all generative models rely on generative deep learning, we refer here to those that build on artificial neural networks\textsuperscript{1}. Given a data distribution $P$, a generative model will model an approximate distribution $P'$. The parameters for the approximate distribution can be learned by an artificial neural network. This learning task is tackled differently by different architectures and training schemes. E.g., autoencoders (Rumelhart, Hinton, and Williams, 1985) and variational autoencoders (VAE) (Kingma and Welling, 2013; Rezende, Mohamed, and Wierstra, 2014) learn to approximate the data through reconstruction via an encoding and a decoding network, while generative adversarial networks (GAN) (Goodfellow et al., 2014) consists of a generator that is guided by a discriminating network. In most cases, the network learns a mapping from a lower-dimensional latent distribution $X$ to the complex high-dimensional feature

\begin{footnote}{For further reading, a comprehensive overview of generative models is given in Harshvardhan et al. (2020).}
\end{footnote}
space of a domain. The model, thus, generates a sample \( p' \) given an input vector \( x \) which should resemble samples drawn from the target distribution \( P \). In the simplest case of a one layer network the generated sample \( p' \) is generated using the function: \( p' = \sigma(Wx + b) \) where \( x \) is the input vector from the latent distribution \( x \in X \), \( \sigma \) is a non-linear activation function, \( W \) and \( b \) are the learned association matrix and bias vector for generating samples in the approximate distribution \( p' \in P' \). The model parameters \( W \) and \( b \) are typically learned through gradient-based optimisation process. In this process, a loss function will require the model to maximise the likelihood of the data either: (i) explicitly, as in the case of autoencoders, autoregressive (Frey et al., 1996) and flow-based generative models (Dinh, Krueger, and Bengio, 2014); (ii) approximately, as is the case in VAEs; (iii) or implicitly, as in the case of GANs. Generative models can also be conditioned on labelled data. In the conditional case, the generative model takes two inputs \( x \) and \( y \), where \( y \) represents the class label vector. Another form of conditional generative models are translation models, such as pix2pix (Isola et al., 2017), that takes a (high dimensional) data distribution as input \( Q \) and learns a mapping to \( P' \) which is an approximation of the true target function \( f : Q \rightarrow P \).

All deep generative models, and in particular ones that generate high dimensional data domains like images, audio and natural language, will have some level of divergence \( D(P||P') \geq 0 \) between the target distribution \( P \) and the approximate distribution \( P' \), because of the complexity and stochasticity inherent in high dimensional data. The goal of all generative models is to minimise that level of divergence, by maximising the likelihood of generating the given data domain. Active divergence methods however, intentionally seek to create a new distribution \( U \) that does not directly approximate a given distribution \( P \), or resemble any other known data distribution. This is either done by seeking to find model parameters \( W^* \) and \( b^* \) (in the single layer case) that generate novel samples \( u = \sigma(W^*x + b^*) \), or by making other kinds of interventions to the chain of computations.

**Survey of Active Divergence Methods**

We present a comprehensive overview and taxonomy of the state of the art in methods for achieving active divergence. In this survey, we will use the term divergence in the statistical sense, as being the distance (or difference) between two distributions. There are other definitions of divergence relevant to research in creativity, such as Guilford’s dimensions of divergent thought (Hocevar, 1980). While there are some parallels that can be drawn between some of the active divergence methods, and theories of divergent thinking; for the clarity of technical exposition, we will be sticking strictly to the statistical definition of divergence in this overview of active divergence methods.

**Novelty search over learned representations**

Methods in this category take existing generative models trained using standard maximum likelihood regimes and then specifically search for the subset of learned representations that do not resemble the training data by systematically sampling from the model\(^2\). Taking account of the fact that any approximate distribution \( P' \) will be somewhat divergent from the true distribution \( P \), these methods seek to find the subset \( U \) of the approximate distribution which is not contained in the true distribution \( U \subset P' \setminus U \not\subset P \). Kazakçı, Mehdi, and Kegl (2016) present an algorithm for searching for novelty in the latent space of a sparse autoencoder trained on the MNIST dataset (LeCun et al., 1998). They start by creating a sample of random noise and by using a Markov chain monte carlo (MCMC) method of iteratively re-encoding the sample through the encoder, then refining the sample until it produces a stable representation. They use this approach to map out all the representations the model can generate, then perform k-means clustering on the latent space encoding of these representations. By disregarding clusters that correspond to real digits, they are left with clusters of representations of digits that do not exist in the original data distribution. It has been argued that these ‘spurious samples’ are the inevitable outcome of generative models that learn to generalise from given data distributions (Kegl, Cherti, and Kazakci, 2018) and that there is a trade off between the ability to generalise to every mode in the dataset and the ratio of spurious samples in the resulting distribution.

**Novelty generation from an inspiring set**

The methods in this section train a model from scratch using a training dataset, but do not attempt to model the data directly, rather using it as reference material to draw inspiration from. We therefore refer to this training set (the given distribution \( P \)) as the inspiring set (Ritchie, 2007).

An approach for novel glyph generation utilises a class-conditional generative model trained on the MNIST dataset (LeCun et al., 1998), but in this case they train the model with ‘hold-out classes’ (Cherti, Kegl, and Kazakci, 2017), additional classes that do not exist in the training dataset distribution. These hold-out classes can then sampled during inference, which encapsulate the subset \( U \) of the approximate distribution \( P' \) that is not included in the target distribution \( U \subset P' \setminus U \not\subset P \). These divergent samples can then be generated directly by conditioning the generator with the hold-out class label, without the need for searching the latent space.

An approach that directly generates a new distribution \( U \) from an inspiring set \( P \) is the creative adversarial networks (CAN) algorithm (Elgammal et al., 2017). The algorithm uses the WikiArt dataset (Saleh and Elgammal, 2016), a labelled dataset of paintings classified by ‘style’ (historical art movement). This algorithm draws inspiration from the GAN training procedure (Goodfellow et al., 2014), but adapts it such that the discriminator has to classify real and generated samples by style, and the generator is then optimised to maximise the likelihood of the generated results being classified as ‘artworks’ (samples that fit the training distribution of existing artworks) but maximise their deviation from existing styles in order to produce the novel distribution \( U \).

\(^2\)An overview of methods for sampling generative models is given in White (2016).
Training without data

Training a model from a random initial starting point without any training data, almost certainly guarantees novelty in the resulting generated distribution. Existing approaches to doing this all rely on the dynamics between multiple models to produce emergent behaviours through which novel data distributions can be generated.

Multi-generator dynamics

Broad and Grierson (2019a) present an approach to training generative deep learning models without any training data, by using two generator networks, and relying on the dynamics between them for an open-ended optimisation process. This approach took inspiration from the GAN framework, but instead of a generator mimicking real data, two generators attempt to mimic each other while the discriminator attempts to tell them apart. In order to have some level of diversity in the final results, the two generators are simultaneously trying to produce more colours in the generated output than the other generator network, leading to the generation of two novel, yet closely related distributions $U$ and $V$.

Generation via communication

An alternative approach to generating without data uses a single generator network, and uses the generated distribution $U$ as a channel for communication between two networks, which together learn to generate and classify images that represent numerical and textual information from a range of existing datasets (Simon, 2019). In subsequent work, by constraining the generator with a strong inductive bias for generating line drawings, this approach can be utilised for novel glyph generation (Park, 2020).

Divergent fine-tuning

Divergent fine-tuning methods take pre-trained models that generate an approximate distribution $P'$ and fine-tune the model away from the original training data. This can either be done by optimising on new training data, or by using auxiliary models and custom loss functions. The goal being to find a new set of model parameters that generate a novel distribution $U$, that is significantly divergent from the approximate distribution $P'$ and the original distribution $P$.

Cross domain training

In cross domain training, transfer learning is performed to a pre-trained model that generates the approximate distribution $P'$ and is then trained to approximate the new data distribution $Q$. This transfer learning procedure will eventually lead to the model learning a set of parameters that generate the approximate distribution $Q'$. However, by picking an iteration of the model mid-way through this process, a set of parameters can be found that produced a blend between the two approximate distributions $P'$ and $Q'$, resulting in the producing the novel distribution $U$ (Schultz, 2020a). This method, was discovered by many artists and practitioners independently, who were performing transfer learning with GAN models for training efficiency, but noted that the iterations of the model part-way through produced the most interesting, surprising and sometimes horrifying results (Adler, 2020; Black, 2020; Mariani, 2020; Shane, 2020).

Continual domain shift

Going beyond simply mixing two domains, one approach that gives more opportunity to steer the resulting distribution in the fine-tuning procedure, is to optimise on a domain that is continually shifting. In creating the artworks Strange Fruit (Som, 2020), the artist Mal Som “iterate[s] on the dataset with augmenting, duplicating and looping in generated images from previous ticks” to steer the training of the generator model (Som, 2021). In this process, the target distribution $Q_t$ at step $t$ may contain samples $q_{t-n}$ generated from earlier iterations of the model at any previous time step $t - n$ where $0 < n < t$. Addi-
tionally, the target distribution $Q_t$, may no longer include samples, or may have duplicates of samples $q_{t−n}$ from previous iterations of the target distribution. Using this process, the target distribution can be continually shaped and guided.

This process of modelling a continually shifting domain often leads to the —generally unwanted— phenomenon of mode collapse (Thanh-Tung and Tran, 2020). However, in Som’s practice, this is induced deliberately. After a model has collapsed, Som explores its previous iterations to find the last usable instance right before collapse. Som likens this practice to the artistic technique of defamiliarisation, where common things are presented in unfamiliar ways so audiences can gain new perspectives and see the world differently (Som, 2021).

**Loss hacking** An alternative strategy, is to fine-tune a model without any training data. Instead a loss function is used that directly transforms the approximate distribution $P'$ into a novel distribution $U$ without requiring any other target distribution. Broad, Leymarie, and Grierson (2020) use the frozen weights of the discriminator to directly optimise away from the likelihood of the data, by using the inverse of the adversarial loss function. This process reverses the normal objective of the generator to generate ‘real’ data and instead to generate samples that the discriminator deems to be ‘fake’. By applying this process to a GAN that can produce photo-realistic images of faces, this fine-tuning procedure crosses the uncanny valley in reverse, taking images indistinguishable from real images, and amplifying the uncanniness of the images before eventually leading to mode collapse. In a similar fashion to Som’s practice (see previous sub-section), one instance of the model before mode collapse was hand-selected and a selection of its outputs turned into the series of artworks Being Foiled (Broad, 2020a).

**Infusing external knowledge** By harnessing the learned knowledge of externally trained models, it is possible to fine-tune models to infuse that knowledge to transform the original domain data with characteristics defined using the auxiliary model. Broad and Grierson (2019b) utilise a classifier model $C_{\text{classifier}}$ trained to differentiate between datasets, in conjunction with the frozen weights of the discriminator $D_{\text{frozen}}$, to fine-tune a pre-trained GAN generator model $G$ away from the original distribution and towards a new local minimum defined by the loss function $L$. $L$ is defined as the weighted sum of the two auxiliary models $L = \alpha C_{\text{classifier}}(G(x)) + \beta D_{\text{frozen}}(G(x))$ given the random latent vector $x$, and $\alpha$ and $\beta$ being the hyper-parameters defining the weightings for the two components of the loss function.

The StyleGAN-NADA framework (Gal, 2021) takes advantage of the external knowledge of a contrastive language–image pre-training model (CLIP) (Radford et al., 2021). CLIP has been trained on billions of text and image pairs from the internet and provides a joint-embedding space of both images and text, allowing for similarity estimation of images and text prompts. In StyleGAN-NADA, pretrained StyleGAN2 models (Karras et al., 2020) can be fine-tuned using user-specified text prompts, the CLIP model $C_{\text{clip}}$ is then used to encode the text prompts and the generated samples in order to provide a loss function where the cosine similarity $S$ between the clip function of the text string $t$ and the generated image embedding $G(x)$ given random latent $x$, can be minimised using the loss $L = S(C_{\text{clip}}(t), C_{\text{clip}}(G(x)))$. This training procedure, guides the generator towards infusing characteristics from an unseen domain defined by the user as text prompts.

**Chaining models**

An approach that is widely used by artists who incorporate generative models into their practice, but not well documented in academic literature, is the practice of chaining multiple custom models trained on datasets curated by the artists. The ensembles used will often utilise standard unconditional generative models, such as GANs, in combination with other conditional generative models such as image-to-image translation networks, such as pix2pix (Isola et al., 2017) and CycleGAN (Zhu et al., 2017), along with other approaches for altering the aesthetic outcomes of results such as style transfer (Gatys, Ecker, and Bethge, 2016). Artists will often train many models on small custom datasets and test out many combinations of different models, with the aim of finding a configuration that produces unique and expressive results. The artist Helena Sarin will often chain multiple CycleGAN models into one ensemble, and will reuse training data during inference, as the goal of this practice “is not generalization, my goal is to create appealing art” (Sarin, 2018). The artist Derrick Schultz draws parallels between the practice of chaining models and Robin Sloan’s concept of ‘flip-flopping’ (Schultz, 2021), where creative outcomes can be achieved by “pushing a work of art or craft from the physical world to the digital world and back, often more than once” (Sloan, 2012).

**Network bending**

Network bending (Broad, Leymarie, and Grierson, 2021) is a framework that allows for active divergence using individual pre-trained models without making any changes to the weights or topology of the model. Instead, additional layers that implement standard image filters are inserted into the computational graph of a model and applied during inference to the activation maps of the convolutional features. As the computational graph of the model has been altered, the model which previously generated samples from the approximate distribution $P'$, now produces novel samples from the new distribution $U$, without any changes being made to the parameters of the model. In the simplest case of a two layer model an association weight matrix $W_l$ and bias $b_l$ vector for each layer $l$. Which generates sample $p_l' = \sigma(W_l(\sigma(W_l x + b_l)) + b_2)$ from input vector $x$ and using a non-linear activation function $\sigma$. In the network bending framework, a deterministic function $f$ (controlled by the parameter $y$) is inserted into the computational graph of the model and applied to the internal activations of the model $u = \sigma(W_l(f(\sigma(W_l x + b_l)), y)) + b_2$, allowing the model to produce new samples $u$ from the new distribution $P'$.

---

3Inserting filters into GANs was also developed independently in the Matlab StyleGAN playground (Pinkney, 2020c).
Beyond the simplest case of a transformation being applied to all features in a layer, the transformation layer can also be applied to a random sub-section of features, or to a pre-selected set of features. Broad, Leymarie, and Grierson (2021) present a clustering algorithm, that in an unsupervised fashion, groups together sets of features within a layer based on the spatial similarity of their activation maps. This clustering algorithm is capable of finding sets of features responsible for the generation of various semantically meaningful components of the generated output across the network (and semantic) hierarchy, which can then be manipulated in tandem allowing for semantic manipulation of the internal representations of the generative model.

In addition to applying filters to the activation maps, it is also possible to enlarge samples by increasing the size of the activation maps and interpolating and tiling them (Pouliot, 2020). The network bending framework has been extended into the domain of audio synthesis (McCallum and Yee-King, 2020) where it has been applied to neural vocoder models using the differential digital signal processing (DDSP) approach (Engel et al., 2020). In order to adapt the framework for the audio domain, McCallum and Yee-King (2020) implement a number of filters that operate in the time domain, such as oscillators. Network bending has also been applied in the domain of audio-reactive visual synthesis using generative models (Brouwer, 2020), with the deterministic transformations being controlled automatically using features extracted from audio analysis.

**Network blending**

Blending multiple models trained on different dataset allows for more control over the combination of learned features from different domains. This can either be done by blending the predictions of the models, or by blending the parameters of the models themselves.

**Blending model predictions** Akten and Grierson (2016) present an interactive tool for text generation allowing for the realtime blending of the predicted outputs of an ensemble of long-short term memory network (LSTM) models (Hochreiter and Schmidhuber, 1997) trained to perform next character prediction from different text sources. A graphical user interface allows the user to dynamically shift the mixture weights for the weighted sum for the predictions of all of the models in the ensemble, prior to the one hot vector encoding which is used to determine the final predicted character value.

**Blending model parameters** A number of approaches, all demonstrated with StyleGAN2 (Karras et al., 2020), take advantage of the large number of pre-trained models that have been shared on the internet (Pinkney, 2020b). Of these almost all have been transfer-learned from the official model weights trained on the Flickr-Faces High Quality (FFHQ) dataset. It has been shown that the parameters of models transfer-learned \( p_{\text{transfer}} \) from the same original source \( p_{\text{base}} \) share commonalities in the way their weights are structured. This makes it possible to meaningfully interpolate between the parameters of the models directly (Aydao, 2020).

By using an interpolation weighting \( \alpha \), it is possible to control the interpolation for the creation of a set of parameters

\[
p_{\text{interp}} = (1 - \alpha) p_{\text{base}} + \alpha p_{\text{transfer}}.
\]

Layers can also be swapped from one model to another (Pinkney and Adler, 2020), allowing the combination of higher level features of one model with lower level features of another. This layer swapping technique was used to make the popular ‘toonification’ method, which can be used to find the corresponding sample to a real photograph of a person in a Disney-Pixar-esque ‘toonified’ model, simply by sampling from the same latent vector that has been found as the closest match to the person in FFHQ latent space (Abdal, Qin, and Wonka, 2019). A generalised approach that combines both weight interpolation and layer-swapping methods for multiple models, uses a cascade of different weightings of interpolation for the various layers of the model (Arfafax, 2020).

Colton (2021) presents an evolutionary approach for exploring and finding effective and customisable neural style transfer blends. Upwards of 1000 neural style transfer models trained on 1-10 style images each, can be blended through model interpolation, using an interface that is controlled by the user. MAP-Elites (Mouret and Clune, 2015) in combination with a fitness function calculated using the output from a ResNet model (He et al., 2016) were used in evolutionary searches for optimal neural style transfer blends.

**Model rewriting**

Model rewriting encompasses approaches where either the weights or network topology are altered in a targeted way, through manual intervention or by using some form of heuristic based optimisation algorithm.

**Stochastic rewriting** To create the series of artworks *Neural Glitch* the artist Mario Klingemann randomly altered, deleted or exchanged the trained weights of pre-trained GANs (Klingemann, 2018). In a similar fashion, the convolutional layer reconnection technique (Růžička, 2020) randomly swaps convolutional features within layers of pre-trained GANs. This technique is applied in the *Remixing AIs* audiovisual synthesis framework (Collins, Růžička, and Grierson, 2020).

**Targeted rewriting** Bau et al. (2020) present a targeted approach to model rewriting. Here, a sample is taken from the model and manipulated using standard image editing techniques (referred to as a ‘copy-paste’ interface). Once the sample has been altered corresponding to the desired goal (such as removing watermarks from the image, or getting horses to wear hats), a process of constrained optimisation is performed. All of the layers but one are frozen, and the weights of that layer are updated using gradient descent optimisation until the generated sample matches the new target. After this optimisation process is complete, the weights of the model are modified such that the targeted change becomes present in all the samples that the model generates.

The CombiNets framework (Guzdial and Riedl, 2018), informed by prior research in combinational creativity (Boden, 2004), can be utilised to create a new model by combining...
parameters from a number of pre-trained models in a targeted fashion. The parameters of existing models are recombined to take into account a new mode of generation that was not present in the training data (an example given would be a unicorn for a model trained on photographs of non-mythical beings). In this framework, a small number of new samples is provided (not enough to train a model directly) and then heuristic search is used to recombine parameters from existing models to account for this new mode of generation.

Further Demarcations
In this section, we highlight demarcations that can be used to classify methods for active divergence. The following categories serve as criteria for further discussion and method comparison.

Training from scratch vs. using pre-trained models
Finding stable, effective ways of training generative models, in particular GANs, is difficult and, depending on the training scheme, there are only a handful of methods that have been found to work successfully. Few methods for active divergence train a model completely from scratch. Instead, most take pre-trained models as their starting point for interventions. This way, training from scratch can be avoided, but fine-tuning may still be required.

Utilising data vs. dataless approaches
Most of the approaches described utilise data in some way, whether as an inspiring set for novelty generation, or for combining features from different datasets (divergent fine-tuning, network blending and chaining models). Even methods for model rewriting use very small amounts of example data to guide optimisation algorithms that alter the model weights. However, methods like network bending, show how models can be analysed in ways that don’t rely on any data, and are used for intelligent manipulation of the models—an approach which could be applied to other methods like model rewriting. Methods that train and fine-tune models without data also show how auxiliary networks and the dynamics between models can be utilised for achieving active divergence.

Human direction vs. creative autonomy
Very few of the approaches described have been developed with the expressed intention of handing over creative agency to the systems themselves. Most of the methods have been developed by artists or researchers in order to allow people to manipulate, experiment with and explore the unintended uses of these models for creative expression. However, the methods described that are currently designed for, or rely on a high degree of human curation and intervention, could easily be adapted and used in co-creative or autonomous creative systems in the future (Berns et al., 2021).

Applications of Active Divergence
In this section we outline some of the applications for active divergence methods.

Novelty generation
Generative deep learning techniques are capable of generalisation, such that they can produce new artefacts of high typicality and value, but are rarely capable of producing novel outputs that do not resemble the training data. Active divergence techniques play an important role in getting generative deep learning systems to generate truly novel artefacts, especially when there may be limited or even no data to draw from.

Creativity support and co-creation
Some of the frameworks presented are already explicitly designed as creativity support tools, such as the network bending framework, designed to allow for expressive manipulation of deep generative models. The Style Done Quick (Colton, 2021) application where many style transfer models have been evolved, was built as a casual creator application (Compton and Mateas, 2015). Though many of the other methods described are still preliminary artistic and research experiments, there is a lot of potential for these methods to become better understood and eventually adapted and applied in more easily accessible creativity support tools and co-creation frameworks.

Knowledge recombination
Reusing and recombining knowledge in efficient ways is an important use-case of active divergence methods. While impressive generalisation can be ascertained from extremely large models trained on corpuses extracted from large portions of the internet (Ramesh et al., 2021), this is out of the capabilities for all but a handful of large tech companies. Instead of relying on ever expanding computational resources, active divergence methods allow for the recombination of styles, aesthetic characteristics and higher level concepts in a much more efficient fashion. Methods like chaining models, network blending and model rewriting offer alternatives routes to achieving flexible knowledge recombination and generalisation to unseen domains without the need for extremely large models or data sources.

Unseen domain adaptation
Active divergence methods allow for the possibility of adapting to and exploring unseen domains, for which there is little to no data available. The network blending approach presented by Pinkney and Adler (2020) can be used for the translation of faces while maintaining recognisable identity into a completely synthesised data domain, something which would not be possible with standard techniques for image translation (Zhu et al., 2017).

The model rewriting and network bending approaches offer the possibility of reusing and manipulating existing knowledge in a controlled fashion to create new data from a small number of given examples, or theoretically without any prior examples if external knowledge sources are integrated, as discussed further below. This approach could also be utilised by agents looking to explore hypothetical situations, by reorganising learned knowledge from world models (Ha and Schmidhuber, 2018) to explore hypothetical situations or relations.
A benchmark for creativity

Generative models represent large knowledge bases that can produce high quality artefacts. There is a lot of unexplored potential for how the information and relationships they contain can be reused and rewritten with frameworks for manipulating them such as network bending and model rewriting. Active divergence frameworks could make good candidates for exploring and evaluating modes of creativity, such as combinational creativity (Boden, 2004) and conceptual blending (Fauconnier and Turner, 2008). These could be used to inform how the features in the model could be re-organised, and then evaluated by examining the artefacts generated from the altered models.

Future Research Directions

In this section we discuss possible future research directions and applications for developing, evaluating and utilising methods for active divergence.

Metrics for quantitative evaluation

For the advancement of research on active divergence, methods for quantitative evaluation will be critical in order to keep track of progress, to compare techniques and for benchmarking. Metrics for active divergence will have to go beyond measuring the similarity or dissimilarity between distributions, as is usually done in the evaluation of generative models (Gretton, Sutherland, and Jitkrittum, 2019). Active divergence metrics should contribute to a better understanding of how the distributions diverge. Therefore, various changes to the modelled distribution should be taken into consideration when looking to measure divergence between distributions in creative contexts. These include increases or decreases in diversity, the consistency and concurrency of change across the whole distribution and whether changes primarily effect low or high level features.

Automating qualitative evaluation

In addition to quantitative evaluation, other metrics are needed for evaluating active divergence metrics. In order to rely less on qualitative evaluation for guiding decisions in creating new models, and do this in computational fashion so that these aspects of the process can be handed over to the computational systems. For instance, a recently developed metric for measuring visual indeterminacy (Wang et al., 2020b), which is argued as being one of the key drivers for what people find interesting in GAN generated art (Hertzmann, 2020), could be used for replacing the qualitative evaluation and curation step done by humans. Other metrics that could be used are: novelty metrics (Grace and Maher, 2019), bayesian surprise (Itti and Baldi, 2009), aesthetic evaluation (Galanter, 2012), or measurements for optimal blends between data domains and evaluating the novelty of changes made to semantic relationships.

Inventing new objective functions

None of the methods presented to date that are based on generative deep learning have been capable of inventing their own objective functions. Instead, methods such as creative adversarial networks (Elgammal et al., 2017) rely on hand crafted variations of well established objective functions. This will be one of most challenging future research directions to overcome, as generative deep learning systems rely on a small handful of objectives that result in stable convergence. However, in conjunction with the development of new evaluation metrics, it may be possible to explore whole new categories of objective functions that diverge from existing data representations and produce artefacts of high value.

Utilising external knowledge

Harnessing expert knowledge external to the dataset, which may come from separate domains or symbolic knowledge representations will allow much more flexibility in how generative models are manipulated in combinational creativity (Boden, 2004) and conceptual blending frameworks (Fauconnier and Turner, 2008). Combining research into analysing the semantic purpose and relationship between features, and creating mappings of those to external data sources or knowledge graphs, would allow for more flexibility in controlling techniques which currently rely on human intervention (network bending, model rewriting). This could be adapted to be controlled and manipulated computationally, allowing for some creative decision making to be handed over to the computer.

Formulating and realising intentions

For many of the methods described, a system that could formulate and realise its intentions would have to be capable of sourcing and creating its own dataset. For instance, a system that wants to create a model that generates hybrids between cats and dogs, would have to be capable of collecting data of cats and dogs separately, and then decide to use some method for network blending to get the desired results. Alternatively, utilising external knowledge sources in combination with semantic analysis of features, would allow computational systems more flexibility in generating new models by altering the semantic relationships between features in model rewriting or network bending approaches.

Multi-agent systems

It has been argued the the GAN framework is the simplest example of a multi-agent system (Agüëra y Arcas, 2019), and frameworks such as neural cellular automata (Mordvintsev et al., 2020) offer new possibilities for multi-agent approaches in generative deep learning. The active divergence methods for training without data described in this paper all rely on the dynamics of multiple agents to produce interesting results, but this could be taken much further. It has been argued that art is fundamentally social (Hertzmann, 2021) and exploring more complex social dynamics between agents (Saunders, 2019) could be a fruitful avenue for exploration in the development of these approaches. There is a large body of work in emergent languages from co-operative multi-agent systems (Lazaridou, Peysakhovich, and Baroni, 2017) that could be drawn from in furthering the work in generative multi-agent systems.
Open-ended reinforcement learning

Open-ended reinforcement learning, where there is no set goal (Wang et al., 2020a), offers possibilities for new more autonomous approaches to achieving active divergence. Reinforcement learning has not been discussed in this survey, but has been used in generative settings (Luo, 2020) in nascent research. Reinforcement learning approaches offer many opportunities for frameworks of creativity to be explored that are not available to standard generative deep learning methods, as they take actions in response to their environment, rather than just fitting functions. Paradigms like intrinsic motivation (Shaker, 2016), cooperating or competing with other agents, formulating and acting on intentions are all concepts that conventional generative deep learning systems alone cannot explore, but these paradigms could be explored in open-ended systems utilising reinforcement learning.

Conclusion

We have presented a taxonomy and survey of the state of the art in methods for achieving active divergence from a range of sources, including artistic experiments, creativity support tools and in computational creativity research. Many of these methods represent nascent areas of research and there is a lot of scope for future work utilising them in co-creative and automated creative systems as they overcome a key shortcoming of mainstream generative deep learning approaches, which are unable to diverge from reproducing the training data in creative ways. In addition, we outline a number of the key future research directions needed in order to advance the state of the art for creativity support tools and computationally creative generative deep learning systems.

Acknowledgements

We thank our reviewers for their helpful comments. This work has been supported by UK’s EPSRC Centre for Doctoral Training in Intelligent Games and Game Intelligence (IGGI; grants EP/L015846/1 and EP/S022325/1).

References


Elgammal, A.; Liu, B.; Elhoseiny, M.; and Mazzone, M. 2017. CAN: Creative adversarial networks, generating “art” by learning about styles and deviating from style norms. Proc. 8th International Conference on Computational Creativity.


Som, M. 2021. Personal communication.


Generative Search Engines: Initial Experiments

Simon Colton,1, 2 Amy Smith,1 Sebastian Berns,1 Ryan Murdock3 and Michael Cook 1
1 School of Electronic Engineering and Computer Science, Queen Mary University of London, UK
2 SensiLab, Faculty of IT, Monash University, Melbourne, Australia
3 Department of Psychology, University of Utah, USA

Abstract

Text-to-image generation involves producing an image which somehow reflects a given text prompt. We investigate the creative potential of a novel approach to this task. This employs three neural models working in concert: a generative adversarial network producing images with input latent vectors chosen by a search guided by a pair of models able to assess the appropriateness of a generated image for a text prompt. For evaluation purposes, we re-frame the task to be analogous to Google-like image search and introduce notions of efficiency, fidelity, variety, sophistication and coherence in the generated images. We have found the approach remarkably successful and explore here its potential for various creative tasks. We propose two approaches to increase efficiency in the generative process, and we evaluate the approach in an experiment simulating commercial design usage. We further suggest ways in which a generative search engine could be used in videogame design via standard usages and via an always-on modality for continuous creativity.

Introduction

As in many areas of AI research and practice, deep learning is making advances for computational creativity applications that may have seemed impossible a few years ago. This is certainly the case in text-to-image synthesis tasks, where images are generated that somehow reflect a user-given text prompt [1]. The main application of this has so far been to produce images which reflect content prescribed by a text prompt, e.g., “a yellow bird in a tree”, aiming to produce photorealistic images which could have been photographed in physical reality. However, there has also been progress in generating images for more abstract prompts such as “the devil in my head”, where images don’t exist in reality and have to be imagined, and likewise artistic or visual prompts such as “a skyline in the style of Mondrian”.

In the next section, we describe a new set of text-to-image techniques that have shown much promise for use in various creative projects. These employ a generative adversarial network (GAN), producing images with input latent vectors chosen by a search guided by a pair of models called CLIP. The CLIP models are able to assess the appropriateness of a generated image for a text prompt, hence act as a fitness function for search strategies. We provide some background to these techniques in the next subsection. To provide a context for benchmarking and evaluation, we re-frame the text-to-image task as akin to image retrieval, and so can propose measures based on internet-based search engines like Google image search, as described in the third section below. This enables us to propose five different kinds of creative task that text-to-image generation can currently help with, and we present a preliminary evaluation of the new methods for each kind of task.

Focusing on applications to design inspiration, we propose and validate two methods for improving the speed of the search when multiple images are returned for a single prompt. To further explore the creative potential of the new wave of text-to-image generators, we simulated a scenario whereby a trained artist employed the approach as if it were an internet image search engine, for inspiration in the commercial setting of tattoo design. We report the overall success of this simulation as further evidence of the potential for neural text-to-image generators to drive forward computational creativity research. To supplement this, we further speculate on how generative search engines could help in videogame design, through standard usages and an always-on approach as a team member. We conclude by describing some of the many future directions for this research.

Background

Generative models such as variational autoencoders [22] (VAEs) have made successful inroads into automated image production by synthesising new images (fakes) to look like they come from a given distribution [34]. VAEs work by compressing media such as images into a smaller latent space of vectors in such a way that they can be decompressed to produce an output that resembles the original media. This enables the direct production of latent vectors, e.g., through random means or a search process, that can be decompressed for generative purposes, and numerous artists have used and abused this approach, as described in [2].

A particularly successful technique for developing image generation models involves adversarial training of two models simultaneously, one to generate fake images and one to critique the images in terms of how realistic they look when compared to a corpus of real images [16]. Some of these Generative Adversarial Networks (GANs) have been trained for specific generation of a single type of image, e.g., faces [20], and others have been trained to generate multiple types of images. For instance, the BigGAN model [5] produces images in 1,000 categories, given a random latent vector and a one-hot class vector specifying a required category.

A technique called contrastive learning aims to train two compression models at once, so the latent representations...
of two different, but related, media items are similar. In particular, OpenAI recently made available a pair of pre-trained models called CLIP [27], which encode images and text prompts as latent vectors so that the cosine distance between encodings for (image, text) pairs that are related is smaller than for pairs where the text and image are not related. CLIP was trained over 400 million image/text pairs scraped from the internet, and has captured a broad and deep understanding of correlations between text and images, covering visual elements such as content, mood, texture, pattern, lighting, emotion and genre, as well as individual objects, people (if numerous examples exist on the internet) and artistic styles. A similar effort to produce a contrastive model called ALIGN has been reported by Google in [19].

CLIP-Guided GAN Image Generation

Colab notebooks [3] are free interactive Python programming environments which are connected to a CPU or GPU provided by Google. To increase accessibility, notebooks can be configured to hide code and expose GUI elements such as text boxes and drop-down lists. In recent practical work, a community of deep learning artists and researchers have been sharing code repositories and Colab notebooks which enable users to type in prompts and generate images which reflect the text. These notebooks largely share the same structure, as follows. For a user-given text prompt \( P \):

(i) A pre-trained generative adversarial network (GAN), \( G \), is loaded into memory.

(ii) The CLIP pair of pre-trained models is likewise loaded.

(iii) A randomly-generated latent vector, \( V \), is input into \( G \), and an image, \( I \), is generated.

(iv) The cosine distance between the CLIP-encoded representations of prompt \( P \) and image \( I \) is used in a calculation to estimate the fitness of \( I \) with respect to it reflecting \( P \).

(v) The fitness is used as a loss function, to update \( V \) in a backpropagation search, producing a new image \( I \).

Steps (iii) to (v) are repeated until the user stops the process.

The notebooks differ mostly in terms of the GAN model they use for image generation, and certain (often extensive) technical tweaks required for search success over the latent vectors in each case. Four examples notebooks that we have experimented with are: the Big Sleep [25] which uses BigGAN and was written by Ryan Murdock (fourth author); Aleph [24], which uses the DALL-E generator [28]; Siren [13], which uses the image generator described in [31]; and Aphantasia [12], which uses texture generation via the Lucent PyTorch library [21]. We have found that these notebooks are not particularly robust to change, i.e., while the notebook may still work, a change could mean that the images produced no longer reflect the prompt given, or indeed have any content at all. Note that a technical description of a similar approach called CLIP-GLaSS is given in [14].

We concentrate here on the Big Sleep implementation, where CLIP is used to guide the BigGAN generator. To utilise this in the context of a generative search engine (see below), we copied the Python code from the Colab notebook to a server at our disposal. We also exposed some parameters and made some improvements. In particular, in the original implementation, the fitness of a generated image is calculated as the average over 128 randomly chosen sub-images called cuts. The cuts are chosen according to a normal probability distribution with mean 0.8s and standard deviation 0.3s, where \( s \) is the generated image size, namely 512 × 512 pixels. After some initial experimentation, we determined that the simpler, faster, approach of assessing only the entire image doesn’t work, but the number of cuts can be reduced to improve efficiency. As described below, we exposed to experimentation the number of cuts, as well as the learning rate hyper-parameter for the backpropagation search.

Also described below, we experimented with various halting mechanisms for the search, as the original Big Sleep implementation relies on users stopping the process when they are happy with the output (or want to discard it). We also implemented the ability for users to provide a target image, \( T \), as well as a text prompt, \( P \), and for the fitness of a generated image, \( I \), to be estimated in terms of a weighted sum of the cosine distance between \( I \) and \( P \) and the cosine distance between \( I \) and \( T \). We implemented the ability for users to specify a fixed category for the class vector input to BigGAN, which constrains the image generation, and a way to spawn up to four server processes for a single text prompt, so multiple images can be generated simultaneously.

In general, we have found the output from the implementation to be remarkably good for a range of different prompts requiring the generation of images ranging from mundane and photo-realistic to abstract and artistic. This has been reflected by hundreds of cherry-picked images shared by dozens of users on social networks like Twitter. Example images arising from our very first usage of two prompts (i.e., with no cherry picking) are given in figure 1, and numerous further examples are given in the remainder of the paper. We have been impressed by three elements of the Big Sleep process. Firstly, it was surprising that BigGAN has such enormous generative potential, as it was trained to produce photo-realistic images of hamburgers and dogs, so we didn’t expect it to have points in the latent space corresponding to images appropriately reflecting prompts such as “A holographic skull”, as described in [32] or “The inside of a black hole”, as shown in figure 1.
Secondly, CLIP’s understanding of how well a piece of text and an image match each other is surprisingly broad and deep. As an example of this, when using the name of one the first author of this paper (Simon Colton) as the prompt, it guided BigGAN to produce images of superheroes, because another person with the same name has a modest online presence including images of superheroes. Thirdly, we’ve been surprised by how quickly the backpropagation search can find appropriate latent vectors for BigGAN, starting from a random position, sometimes converging after 50 steps and usually before 200. In particular, researchers in the online art/technology community developing text-to-image Colab notebooks have also informally experimented with using evolutionary approaches to search for latent vectors. Unpublished results indicate that the process was up to three times slower and not nearly as successful – in terms of image quality – as the backpropagation approach.

Generative Search Engines

OpenAI is reserving for commercial exploitation a one-shot text-to-image generation system which has been used for their DALL-E project (openai.com/blog/dall-e). This can take a text prompt, encode it into a latent space and then decode it as an image, without any search required, using the CLIP encoding methods. This, and the pace at which generative deep learning is advancing, points to future one-shot implementations that will be reliable and fast. Hence we can make a meaningful comparison with Google-like internet image retrieval engines, i.e., we can imagine in the near future that standard online image retrieval searches are supplemented with generative search engines which make, rather than retrieve, images. The latter will complement the former by being able to produce images that don’t, or couldn’t, exist in reality, including images that would normally require imagination in people to produce.

In this context, it would seem sensible to evaluate the Big Sleep and other generators in terms of image retrieval [30]. We draw from this literature and from assumptions about normal usage of web search engines to propose the following measures for the success of a generative search engine. In the scenario where a user-given prompt has returned multiple images, we can evaluate the following properties:

• **Efficiency**: how fast the full set of images are generated.

• **Fidelity**: how well the images reflect the prompt in the subjective view of the user.

• **Variety**: how visually or conceptually varied the set of generated images are, in the subjective view of the user.

In addition, while we would expect a Google search to return relatively sophisticated images, we have found that image generation can result in failures because the images are (a) devoid of detail, i.e., blank or roughly patterned images or (b) detailed but too noisy to be interpretable. Moreover, in cases where the image is detailed and interpretable with respect to general impression of the given prompt, sometimes the images are too incoherent to be of any value. Hence, we suggest also considering the following two measures of value for individual generated images:

- **Coherence**: how well the images stand up to scrutiny on detailed inspection.
- **Sophistication**: how detailed, information-rich and interpretable the images are.

Fig. 2 shows images rejected for lack of sophistication, and we return to the question of coherence in the next section.

Creative Usage Scenarios

It is impossible to characterise all the ways a generative search engine might be used in creative projects. To begin to understand the potential, we propose below five different scenarios where a generative search engine might feasibly be employed, followed by a preliminary evaluation.

Artistic exploration

The majority of people using the Colab notebooks have so far done so for artistic and entertainment purposes, largely posting generated images on social media and blog posts, e.g., [29]. There has been much activity and cherry picking of the results, from which an overly successful view of the approach might be gained. That said, we have successfully used a combination of the four notebooks mentioned above for an art project [32], and found the process rewarding. Each different art project engaging the notebooks has differed in terms of the image requirements, and the same will likely be true for generative search engines. In our usage, a high variety of images over the same prompt was always valuable, along with sophistication of the images. However, we were often less concerned with fidelity to the prompt and coherence of the images, as we wanted to encourage imaginative interpretation of the results by audience members. In many cases, we were more interested in how the content of the images portrayed mood and style rather than coherence. Two example images in this vein are given in figure 3.

![Figure 2: Images rejected for lack of sophistication](image1.png)

![Figure 3: Images generated by the Big Sleep Colab notebook for prompts: “A galaxy funfair ride” and “A painting of a cauldron of magic”. Note the word ‘galaxy’ added to image 1, which is not uncommon.](image2.png)
Automated artistic treatment

Building on much work in graphics-based pastiche generation, neural style transfer [15] enables visual answers to questions such as: “If Claude Monet painted exactly this scene, what might it look like?” A next logical step is to ask questions like: “If Claude Monet painted this subject matter, what might it look like?” This is enabled by CLIP-guided GAN image generation, which, due to its random start, can produce multiple treatments of the subject. For instance, to explore an artistic treatment that couldn’t have happened historically, using the prompt “A modernist building in the style of Claude Monet” for our Big Sleep approach, the images in figure 4 were produced. We see there appropriate, novel, buildings in a style/setting which reflects the artist.

Image manipulation

As mentioned above, we enabled the process to take a target image and a text prompt – as pioneered by artist Mario Klingemann¹ – in order to generate images which look like the target, but also reflect the text. As examples, in figure 5, target images of two faces were given, along with three text prompts. Equal weighting of similarity to the image and the prompt was specified. We see that each generated image resembles the subject matter (person) and reflects the text prompt. Also given in figure 5 are some generated images starting from a photograph with text prompts “In the style of X” where X is a well known artist. This is another application of automated artistic treatment as above, and we see that, unlike with standard style transfer, while the subject matter is per the original digital photo, the scene portrayed varies from it. This helps to extend the generation of pastiche images in an artist’s style to include content and scene arrangement, rather than just transferring textures, colours and abstractions onto a given digital photo as in [15].

Imaginative idea visualisation

While stock photography and internet searches will satisfy many design needs in terms of photos of scenes from reality and artistic images of imagined realities, they will never be able to satisfy every visual scene that someone might want. For instance, the internet search prompt: “Church in an eyeball” produces no useful results from a Google image search. Text-to-image generative search engines can be used for these kinds of tasks, and many highly imaginative scenes have been visualised using the Colab notebooks and shared on social media. As an example, using our version of the Big Sleep, the images in figure 6 were returned for “Church in an eyeball”. As another example, the prompt “An iPhone in the snow” produced an image where the iPhone was stood up, as if the snow was a dock for it. Such visualisation techniques could be highly valuable during creative ideation stages in, for example, advertising and other creative industries.

Design inspiration

In certain scenarios, the generative search reliably produces highly varied, coherent and sophisticated images with high fidelity. In such cases with high reliability, the generated images could be mined for design inspiration, with ideas being extracted directly from the imagery produced. Design inspiration tasks undertake imaginative idea visualisation as above, hence design inspiration extends idea visualisation. The difference comes in terms of the coherence of the images produced: if the images are coherent enough then they can be examined and may provide design inspiration; if not, then only a general impression can be gained.

¹twitter.com/quasimondo/status/1353300845266411521

Figure 4: Images generated for the prompt “A modernist building in the style of Claude Monet”.

Figure 5: Top two lines: Target image, text prompts and generated images. Bottom two lines: target image, artistic style prompt and generated images.

Figure 6: Generated images for “Church in an eyeball.”
A Preliminary Evaluation

We plan to investigate fully the potential of generative search engines in each of the above usage scenarios and more, but we concentrate in the next two sections on design inspiration projects. As an indication of the value of the approach for design inspiration, the high-quality images in figure 7 were all produced in a batch of 100 using the same text prompt.

For the artistic exploration usage, it is clear that the new methods work and are having an impact, as evidenced by the sharing of hundreds of generated artistic images in blog posts, NFT platforms and social media, and in research papers like [32] since the notebooks were published in early 2021. In these cases, images failing all but the sophistication measure could be considered useful in art projects. That is, incoherent images and those not matching the prompt very well could be presented as more abstract pieces requiring interpretation, and users seem prepared to cherry pick from what may be a small yield with low variety.

For the automated artistic treatment usage, image manipulation and idea visualisation scenarios, we tested 50 different text prompts and 50 target-image/prompt pairs which we felt were representative of the kind of things searched for, with examples given in figure 5. For each prompt, we generated 25 images using 300 search steps and the standard Big Sleep setup. We recorded a curation coefficient – defined informally in [7] as the proportion of images we were prepared to show people – and reasons for failures. We found:

• When producing artistic treatment images, we tried prompts covering objects, buildings and people, and a dozen visual artists with distinctive styles, such as Picasso, Monet, de Lempicka and O’Keefe. We found that the process often failed because the design problem was too hard, and report a curation coefficient of only around 20%. The process is most successful for famous artists like Picasso, who has many different styles including abstract and cubist treatments. This is because (a) the fame (hence much coverage in the 400 million internet training examples) and variety of styles means that CLIP has multiple options for guiding the search to a local minima, depending on the random starting point for the latent vector, and (b) unusual scene constructions – in the sense of representing physical reality – are associated with cubism and abstraction, hence jumbled and incoherent images, which are often generated, are fine.

In other cases, with higher realism in the scenery, for instance with Tamara de Lempicka, work less well, as the jumbled generated content lowers the values of the images, even if the painting style is transferred. To optimise the fidelity of this approach, further study of how CLIP has entangled subject matter and style (e.g., portraits and cars versus rich colours in the art of de Lempicka), when learning about the visual properties of artists’ works, will be required. This may lead to automated prompt engineering of the text that is used to guide the search, in order to improve matters. Another approach is to split the generation into two parts, for instance first generating a content image with a suitable prompt and then using the latent vector for this as the starting point, rather than a random vector, with the prompt “In the style of ...” for the artist. If the second search uses a much smaller learning rate, it can tweak the generated image and often produces an appropriately stylised image.

• For imaginative idea visualisation, there is much variation in the fidelity of the output, depending on the nature of the prompt. In particular, there are sweet spots for specificity, coherence and constrainedness of the text prompt, and more study of user expectations for prompt types will be needed to make generative search engines satisfying. For instance, while a prompt such as “a white blackness” is over-constrained in the sense of being logically unachievable, users know this and may be more amenable to any appropriate output. This may similarly be true for prompts such as “Venus in love” that can be satisfied appropriately in numerous, sometimes abstract ways, as the user will have an expectation that the result will require interpretation. On the other hand, for highly specific prompts such as “an orange on top of a lemon on top of a table” there will likely be lower expectation that the results will need interpretation, and users may be more critical of poorly composed results.

We found best cases with a 25% curation coefficient, e.g., for the prompt “A church in a desert” and “A necklace made out of coal”, to worst cases with 0% curation coefficient, e.g., for “A palace with no windows”. The average curation coefficient was around 5%, with most images being rejected due to lack of fidelity and/or lack of coherence.

• The image manipulation scenario was the least satisfying, with an average curation coefficient of around 2%. This is not surprising given that the constrainedness of the problem is doubled when both a text prompt and a target image are used. We only rarely found outputs which looked like the target and reflected the prompt, although when this did happen, the images were surprisingly good. Fortunately, there are other approaches to GAN-based image manipulation which could be incorporated into a generative search engine, e.g., using StyleGAN in [26].

Figure 7: Images generated in the same batch of 100 images by our version of the Big Sleep generator for the prompt: “Any architectural work that does not express serenity is an error” (a quote from architect Luis Barrigan).
Improving Efficiency for Design Inspiration

In porting the Big Sleep code from a Colab notebook to a server at our disposal, we took the opportunity to implement some improvements. In particular, as BigGAN was trained on the ImageNet competition images, a random latent vector coupled with a random class vector usually produces an image of a dog or a bird. After the search process, many generated images retained the look of a dog or bird, which becomes tedious very quickly. To improve matters, while keeping the randomness, which is a key to the success of the process, we enabled the search to start with 20 random pairs of latent/class vectors and choose the one with CLIP encoding closest to the CLIP encoding of the text prompt, yet furthest from the CLIP encodings of “dog” and “bird”.

One of the most successful series of prompts have been related to architecture, where generated images have been reliably high in fidelity, coherence, sophistication and variety, as evidenced with images for the same architecture prompt in figure 7. We see that all the images portray serene architecture, yet they are quite varied, and – while not perfect – they have enough coherence and sophistication that architectural elements could potentially be extracted for design inspiration. Given this, we were able to concentrate on improving the generative search engine efficiency in this domain.

Curtailing Search

The fitness of a generated image is calculated as the average of the cosine distances of the CLIP encodings of \( n \) sub-image cuts of it from the CLIP encoding of the text prompt, with \( n = 128 \) in the original implementation. Cosine distance is computed along dimension -1 then scaled by -100, and high quality images tend to range in fitness from around 20 to 50. We’ve also found that 300 iterations improving the latent vector is almost always enough for the search to converge onto an image, and further steps will likely not change matters. Envisioning a situation where natural language processing could be used to analyse and improve prompts, and to choose specialist procedures, we experimented to test such a procedure. In particular, we’ve found that any generated image with architectural content that has fitness 29 or above will be almost guaranteed to have high sophistication and fidelity, and sufficient coherence around 80% of the time. Examples of generated architectural images that we judged to be too incoherent are given in figure 8 – we see that on close inspection, the buildings are too confusing to easily extract design ideas from.

Given the reliability mentioned above, it is possible to test strategies which curtail a search when a generated image reaches the fitness bar of 29. In the setting of a generative search where users expect high quality images returned quickly, we can hypothesise that a fail-fast approach – where searches that are clearly not working are abandoned early – would have benefits. To test this, we took 10 well known quotes from architects, such as “Architecture aims at eternity” (by Christopher Wren) verbatim as text prompts for generative sessions producing 25 images each. We varied the backpropagation learning rate over the values \( \{0.07, 0.09, 0.12, 0.15\} \) and the number of cuts ranged over \( \{32, 50, 64, 80, 100, 128\} \), producing \( 50 \times 4 \times 6 = 1200 \) batches of 25 images. We ran each search for a full 300 iterations, which enabled us to simulate different search-halting strategies to find the most efficient ways to generate a certain number of quality images (on average).

In simulating a generative search engine, we modelled the usage of 1, 4 and 16 GPUs with load-balancing, meaning that each of the 25 image generation processes are run on the next available GPU. We specified that simulated searches should stop immediately and return any image with fitness \( \geq 29 \). We simulated a total limit of number of iterations from the range \( \{150, 200, 250, 300\} \), at which point the best ever (in terms of fitness) image is returned. Moreover, to test fail-fast strategies, we further specified that searches should stop and return no image if the fitness is below \( f \) after \( s \) iterations with \( (s, f) \) ranging over \( \{(10, 15), (30, 20), (50, 25), (70, 27), (90, 28)\} \). These pairings were chosen by examining failed generations leading to clearly low sophistication images.

Depending on the search requirements, it is possible to optimise the search for number of quality \((f \geq 29)\) images returned or speed, or a balance of both. With the parameters exposed during the generation sessions, along with those for the simulation, we were able to select from the 1,920 search setups only those which yielded at least 1, 5 and 10 quality images in every batch. These were then sorted in increasing time taken, and the fastest setups given in table 1. We see that, if speed is the most important factor in the generative search engine, then reducing the cuts and increasing the learning rate causes fails quickly and the session ends in 442s, 118s or 53s for the entire session, run on 1, 4 and 16 GPU setups respectively. The down-side to this is, of course, an expected yield of only 4.2 quality hits per batch. On the other hand, if quality yield is the priority, then increasing the cuts, decreasing the learning rate and scrapping the early abandonment of searches can produce an expected yield of 10 quality images, albeit in slower times of 752s, 214s and 87s. As shown in table 1, the compromise of producing an

<table>
<thead>
<tr>
<th>R</th>
<th>End</th>
<th>Cuts</th>
<th>LR</th>
<th>(s, f)</th>
<th>Hits</th>
<th>1/4/16 GPUs (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>200</td>
<td>64</td>
<td>0.12</td>
<td>(50, 25)</td>
<td>1/4.2/10</td>
<td>442/118/53</td>
</tr>
<tr>
<td>5</td>
<td>300</td>
<td>80</td>
<td>0.09</td>
<td>(30, 20)</td>
<td>7/9.5/10</td>
<td>569/109/74</td>
</tr>
<tr>
<td>10</td>
<td>300</td>
<td>100</td>
<td>0.09</td>
<td>None</td>
<td>10/10/10</td>
<td>752/214/87</td>
</tr>
</tbody>
</table>

Table 1: (R)quired guaranteed quality results in each batch; example search setups, including total steps allowed (End), number of sub-image cuts, learning rate (LR) and \((s, f)\) (fitness) threshold; min/av/max quality results (Hits) and durations for 1, 4 and 16 GPUs with load-balancing.
expected yield of 9.5 quality images in 659s, 109s and 74s is possibly the best setup for the generative search engine.

**Plateau Detection**

As depicted in figure 9, the fitness of a generated image tends to plateau at some stage during the search (or it plummets to negative numbers). In the situation where it's not possible to know in advance what a suitable fitness threshold would be, this plateau can be detected and used to end the search. After some experimentation, we found the following routine for this to be satisfactory: using the numpy `polyfit` package, a line fitting the fitnesses recorded over a window of the previous 30 iterations is calculated, and if the gradient of this line becomes negative, the search stops and the best ever recorded image is returned. Note that the search is not stopped if a plateau is detected before 50 steps, as we found that this occasionally ended the process too soon. In figure 9, the search would have been terminated using this routine at step 192, saving 108 steps, and reducing the time taken from 113 to 78 seconds.

The best image generated (for the prompt “A painting of a realistic greyscale bee”, as per the experiment described below) at this early stopping point and the best over 300 steps are both given in figure 9, and we see that they are nearly identical. Sampling 500 such pairs, we found that only around 10% of pre-plateau best images differed significantly from the best for 300 steps, and many were not always worse. We felt this was acceptable in return for an average speed-up of around 30-40%, depending on the search setup.

**Results**

On average, the participant rejected immediately 10.0 images per batch of 25, ranging from just 1 (for prompts including “A painting of Alice in Wonderland with the Cheshire Cat”) to 22 (for the prompt “A graphic design of a greyscale geometric wolf head”). The participant reported that the main reason for rejection was lack of fidelity to the text prompt. Of the non-rejected images, on average 3.4 per batch provided some design inspiration. Only 3 of the 80 batches contained no inspiring image, e.g., for “A photo of a colourful and fierce female gypsy wearing a headscarf”.

In some cases, as many as 10 of the 25 images provided design inspiration, e.g., for the prompt “A graphic design of an anatomical heart with atmosphere”. An example image for this prompt, chosen by the participant, is given in figure 10. The participant reported the following qualities as providing inspiration: recognisable shapes (as a whole but also within the heart); appropriate use of colour (muted darker blues next to shades of a visceral crimson red); a serene white background (contrasting with the foreground heart adding atmosphere); an innovative angle and perspec-

---

**An Illustrative Design Project**

To further explore the potential of text-to-image generative models for design inspiration and idea visualisation, we ran an experiment with a participant who has studied fine art to degree level, and who has had a career as a professional tattoo artist. The participant provided 8 client specifications for tattoos from past projects, summarised into short phrases:

A wicca-related colourful neo-traditional witch
A realistic greyscale bee
An anatomical heart with atmosphere
Alice in Wonderland with the Cheshire Cat
A greyscale geometric wolf head
A colourful slice of pizza
A colourful and fierce female gypsy wearing a headscarf
A deer head with flowers in a Victorian style frame

Each specification was taken verbatim as a text prompt for image generation using the plateau detection approach described above, and 25 images were generated. We also prefixed each of the following nine phrases to produce more text prompts: “A graphic design of”, “A tattoo of”, “An emoji of”, “An illustration of”, “A painting of”, “An icon of”, “A line drawing of”, “A photo of” and “A sketch of”. For each of the prefixed prompts, we generated 25 images, hence $8 \times 25 = 200$ images were produced in total. For each of the 80 batches of 25 images reflecting a prompt, the participant was then asked to assess (a) the number of images to reject immediately for reasons of fidelity, coherence or sophistication (b) of the remaining, how many provided some inspiration for a potential tattoo design, and (c) a score for diversity from 1 (low diversity) to 10 (high diversity). The participant also highlighted the kinds of visual and conceptual qualities in the images that might be useful for the task of tattoo design. Finally, the participant chose an image and produced a tattoo design inspired by it.

---

Figure 9: Fitness increase during search. First image: from start of the plateau (at dotted line). Second image: best seen.

Figure 10: Images with suitable qualities for design inspiration. First prompt: “A graphic design of an anatomical heart with atmosphere”. Second prompt: “An illustration of a wicca-related colourful neo-traditional witch.”
tive for the heart; the overall composition of the image (with the focal element taking up roughly two thirds of the space, adhering well to the rule of thirds); and the unprompted addition of a butterfly wing at the top of the image.

The participant chose this image to produce a final tattoo design for, as portrayed in figure 11, and we see that the butterfly wing has been incorporated. We expect such serendipitous effect to be common with generative search engines. The second image in figure 10 was generated for the prompt “An illustration of a wicca-related colourful neo-traditional witch”. The participant reported favourably: an interesting composition (witch’s pose, ratio of ground to sky), saturated fantasy themed colours, appropriate background (a suggestion of being outside in nature), texture and contrast.

Adding prefixes to the prompts is a form of prompt engineering which could in principle be done automatically to improve the quality of outputs. We found that the average number of rejects for the unadulterated prefixes was 4.125 and the average number of images providing design inspiration was 9.4. This baseline was improved upon by four of the prefixes, namely “graphic design”, “illustration”, “painting” and “sketch”. The highest average yield of inspiring images (11.6 per 25) came from the “graphic design” prefix, and the lowest average number of rejects (4 per 25) came from “painting”. The worst performing prefix was “line drawing”, with, on average, 1.6 inspiring and 15.4 rejects per 25 images. The participant reported overall satisfaction with the process, highlighting the biggest differentiator for success/failure being the nature of the scene specified by the text prompt. Their impression was that too many images were too low in coherence for tattoo design, but when they were coherent, they were often genuinely inspiring.

Potential Applications to Videogame Design

The games industry represents a variety of exciting application areas for generative search engines. Games are complex to design and develop, requiring a wide variety of skills which leaves many smaller teams or individuals in need of creativity support tools. At the same time, large development studios need tools to help mitigate risk, support experimentation, and help large creative teams convey ideas to one another. We believe generative search engines can contribute to solving all of these problems.

Generative search engines can be applied to many of these use cases through standard prompt-and-curate processes similar to those we have described earlier in this paper. For instance, it is common to produce concept art and explore visual styles during a game’s early pre-production. This can draw inspiration from many sources, and is usually highly freeform. Generative search engines could play a traditional role here in imaginative idea visualisation or even design inspiration, using prompts based on early design goals. As an example, we could imagine a design team working on a game that requires imagery inspired by the game Dark Souls, and trying the prompt “a mountain-top castle in the style of dark souls”. Given that Dark Souls imagery was probably in the 400 million images used to train it, CLIP was able to guide BigGAN to produce a striking image when we used this prompt, as portrayed in figure 12. The compare, the second image in figure 12 is a real piece of concept art produced in pre-production for Dark Souls.

Many games are often made available to players while in development, as so-called ‘early access’ or beta versions. Such games often use placeholder art to make the experience feel more complete while the game content and design are finalised. This happened with games such as DOTA 2 or Slay The Spire. In experiments using suitable prompts, our generative search engine was able to produce images which match or exceed the quality of such placeholder content, and would

![Figure 12: Left: An image generated for the prompt “a mountain-top castle in the style of dark souls”. Right: Concept art for the game Dark Souls.](image12)

Figure 13: A card depicting a flaming sword from the game Slay The Spire. Left: placeholder art. Centre: a generated mock-up created using the prompt “A fantasy sword enchanted with fire magic”. Right: final in-game art.
be ideal for smaller art assets for things such as skill icons, user interface elements, or even character portraits. Figure 13 shows an example of placeholder art from the game *Stay The Spire*, along with an alternative placeholder using generated art as contrast, shown in the centre of the figure. Given our early impressions producing 100 images for each of 16 game-related prompts, we estimate curation rates for such applications would vary depending on the type of content required, from between 20% to close to 80% acceptable.

These straightforward applications of generative search engines may miss an opportunity to encourage different ways of engaging with these systems. We believe that such generative search engines could be set up to act as continuously creative AI agents, that function less like tools and more like co-workers [10]. In this setting, a generative search engine would be set up within a studio to continuously create new artworks based on prompts it receives from multiple sources. Some of these sources might be explicit – allowing any employee to add a prompt to a queue, for example. Other sources could be sought out by the system itself, through means such as studying changes made to game design documents or new features added to the game.

As an example, the art team working on pre-production of a new game might begin discussing possible directions for the art style and record notes on their meeting discussions citing sources of inspiration, or descriptions of styles they are interested in. Once these notes are uploaded and shared to the studio’s servers, the always-on generative search engine identifies potential prompts from within this document, such as “a coloring book world full of vibrant characters”, and produce some sample images that the art team can see and include in their planning and concept work. Figure 14 shows a series of example images created from this prompt, which is based on the top-level description of the game *Chicory*. These images are inspirational, suggesting colour palettes as well as form, texture and linework styles.

We believe that using generative search engines in such a way could transform our relationship with them. By re-framing the system from a time-constrained tool to a creative collaborator, we shift expectations of the system as well as increasing its ability to surprise people and contribute meaningfully to active discussions. Such an approach may also alleviate problems such as curation, as since the system is working passively, its unused output can more easily be skimmed or ignored.

**Figure 14: Image generated for prompts related to *Chicory*, a vibrant game about colouring in the world, e.g., “a coloring book world full of vibrant characters”.**

**Conclusions and Future Work**

The CLIP pre-trained models were released in early 2021, and there has already been an explosion of uses, including in text-to-image projects such as those described here. We have attempted to assess the potential value and impact of CLIP-guided GAN image generation, by looking at an implementation using CLIP to guide BigGAN. To the best of our knowledge we are the first to suggest switching context from text-to-image Colab notebooks to a generative search engine. This has enabled us to introduce appropriate measures of value, and increased efficiency with fail-fast and plateau-based stopping mechanisms. In an experiment designed to estimate the potential of generative search engines for design inspiration, a trained artist reported overall satisfaction and many benefits to the approach for creative design projects. We also highlighted some potential use-cases and interaction modalities for videogame design.

Visual imagination models [4], visual blending [11] and text-to-image approaches have been studied from a computational creativity perspective, e.g., text-to-emoji [33], newspaper text-to-collage [9] and poetry text-to-visual interpretations [17] have been investigated. With the increase in quality and fidelity afforded by automated search for GAN inputs, we expect them to be used in other computational creativity projects. One such area is fictional ideation [23], where automated invention of fictional ideas could be extended with automated visualisation of those ideas. In general, we expect to see much research into automated prompt engineering which can take a user’s raw text and produce versions that increase the value of the images generated.

As with many deep learning techniques, such as human-steerable GANs [18], the methods described here can highlight biases inherent in pre-trained models such as CLIP, which must be catered for in generative search engines. For instance, Gustave Coral posted images of faces originally generated by CLIP-guided StyleGAN [20]. The prompt “Poor Man” produced the face of a person of colour, while “Rich Man” produced a white face (twitter.com/gustavecoral/status/1349826749404749824). We have experienced this ourselves, e.g., in figure 5, an image of a female face with glasses was produced in response to the prompt “really angry”, which may highlight a bias.

We fully expect generative search engines to complement stock photography and internet image retrieval in creative industry practice in the short to medium term. To achieve this, improvements in the underlying generative processes will be needed, for instance training GANs specifically for the purpose of outputting imaginative visual imagery. Improvements in contrastive image/text learning and in one-shot image generation will also be required. It will eventually become as common to search for images that don’t exist yet, as it currently is to search for images that do exist, and to expect high quality, on-point images to be returned. We believe that this will revolutionise the creative industries and provide many avenues for research in computational creativity. In particular, we are planning for The Painting Fool computational creativity system [6] to use a generative search engine in order to express aspects of its daily existence, as per the notion of the machine condition, described in [8].
Acknowledgements
We would like to thank the anonymous reviewers for their very helpful and thorough comments. We would also like to thank the ICCC’21 organisers for allowing camera ready versions of papers to have two extra pages, which helped a lot. Amy Smith and Sebastian Berns are funded by the EPSRC Centre for Doctoral Training in Intelligent Games and Games Intelligence (IGGI) [EP/S022325/1].

References
8. Visual Creativity
Surprising image compositions

Othman Sbai  
Ecole des Ponts ParisTech  
othman.sbai@enpc.fr

Mathieu Aubry  
Ecole des Ponts ParisTech  
mathieu.aubry@enpc.fr

Camille Couprie  
Facebook AI Research  
coupriec@fb.com

Abstract

Visual blending is a powerful and effective communication tool used in advertising, news and art. By taking objects out of their natural context, and by leveraging visual, linguistic, or phonetic analogies, artists create surprising and challenging composite images. Building on recent image retrieval, completion and composition methods, we design an automatic tool for the creation of visual blends through object replacement grounded in perceptual and semantic features. Given a selected object in an image, our model searches for visually similar but semantically different objects and performs the image blending automatically, leading to surprising image combinations. Using an automatic metric and a human study, we test our composition method with different foreground search approaches and show the potential of this novel artistic tool.

Introduction

Visual metaphors are composite images obtained by blending objects that share a given analogy from different images. They are commonly used in advertising, news and art (Gkiouzepas and Hogg 2011; Phillips and McQuarrie 2004; Forceville 1994). In fact, (Jeong 2008) show that visual advertisement are much more persuasive when based on visual metaphors. While they are sometimes hard to decode (Petridis and Chilton 2019), it is even more challenging to obtain image compositions that have a strong conceptual grounding (Cunha, Martins, and Machado 2020). The collage creation process can be tedious as it not only requires the artist to find a new interesting analogy idea but it also involves a lengthy process of image search and image blending. In this work, we leverage recent visual object retrieval and image composition advances to improve the collage creation experience by suggesting varied combinations given a selected input object. Examples of obtained compositions are shown in Fig. 1. Since our approach grounds the compositions using perceptual features and semantic ones only, the obtained visual blends cannot be qualified as visual metaphors. A typical use case for the method we propose would be an interactive setup, where the artist selects the object of interest. Then, our algorithm automatically searches for visually similar but semantically different foregrounds in a given database of images and performs the object’s copy-paste seamlessly, thus suggesting interesting visual blends.

Our contributions are two-fold. First, we design a foreground image search strategy adapted to the real-time setting that suggests interesting foreground combinations based on the local features similarity with the query foreground object. In particular, we experimentally study the trade-off between the quality of the composite image and the surprising aspect of the composition. Second, we propose a simple copy pasting model that performs geometric and color adaptations to the foreground object in addition to image inpainting. Our composition network is easier to train than competing methods, relying solely on supervised training on synthetic images, but proves to be robust and effective. Moreover, because our geometric and color transformations are affine transformations, they can be applied to images of any resolution, and easily used as an initialization for a manual refinement in a standard editing software.

Related Work

Visual blends creation. Many works have addressed the challenge of visual conceptual blends creation. (Steinbrück 2013) describes a method based on geometrical shape correspondence and object semantics to replace objects with new retrieved ones. Similarly, (Chilton, Petridis, and
Agrawala 2019) proposes a shape based algorithm for finding and matching objects to blend together with a handcrafted blending synthesis method. (Xiao, Linkola, and others 2015) presents Vismantic, a framework for generating image compositions based on a textual input, in order to express a specific meaning, using semantic associations and basic visual operations such as juxtaposition, replacement or fusion. (Tendulkar et al. 2019) propose an approach to make text visually appealing by replacing individual letters with cliparts relevant to a theme and which visually resemble the letters. Recently, (Cunha, Martins, and Machado 2020) provides a roadmap for generating visual blends. They highlight important steps for the conceptualisation of the generated composite by grounding it using perceptual, naming/homophones or affordance attributes. However, they do not provide an implementation of that framework. Simultaneously to our work, (Ge and Parikh 2021) uses adversarial learning to train models generate text based visual blends. In contrast to these methods, we do not use textual input to create visual compositions, but use both perceptual similarity and objects semantics to suggest relevant compositions. Similar to our work, (Karimi et al. 2018) proposes a system for creative ideation through the exploration of conceptual shifts using sketch similarity to find similar sketches from different categories. Also, (Cunha, Martins, and Machado 2018) present a visual blending system for emoji generation, by searching for related concepts and emojis based on semantic data and blend them using juxtaposition, replacement or fusion as in (Xiao, Linkola, and others 2015). Instead, we propose a new image composition method based on recent advances in visual blending methods that allows us creating realistic composites from natural images.

Searching for relevant objects to replace an existing one have been tackled in other works without the aim of creating visual metaphors. (Tsai et al. 2016) presents a pipeline for sky replacement to search for proper skies and perform a semantic-aware color transfer. (Chen et al. 2009) constructs a photomontage from a sketch by searching for candidate images matching the provided text label and performing the composition. (Zhao et al. 2019) instead searches for foreground objects that are semantically compatible with a background image given the category of the object to find.

**Image blending.** Early works on automatic image composition (Burt and Adelson 1983; Milgram 1975) use a multi-resolution image representation to create large mosaics of images. The seminal work of Poisson image blending (Pérez, Gangnet, and Blake 2003) proposes an elegant mathematical formulation based on solving Poisson equations to seamlessly blend images in the gradient domain. Several works improved the Poisson blending approach (Jia et al. 2006; Tao, Johnson, and Paris 2010), which remains a very strong baseline for image composition.

Another line of work have tackled reducing the color discrepancy between composited images. Traditional image harmonization methods focused on better matching low level statistics between source and target images (Xue et al. 2012; Lalonde and Efros 2007). (Xue et al. 2012) identifies image statistics that are correlated with composite realism such as luminance, saturation, contrast, while (Lalonde and Efros 2007) studies color statistics on a large dataset of realistic and unrealistic images to improve composites and discriminate unrealistic ones. More recently, color harmonization (Cohen-Or et al. 2006) can be performed using deep learning methods (Yan et al. 2015; Tsai et al. 2017; Cun and Pun 2020) that learn appearance adjustment using end-to-end networks. (Cong et al. 2020) contributed a large-scale color harmonization dataset and a network to reduce foreground and background color inconsistencies.

In addition to color adjustment, some works study the geometric corrections necessary to place the new object in its new context. Using spatial transformer networks (Jaderberg et al. 2015), a differentiable module for sampling an image through an affine transformed grid, several works such as (Lin et al. 2018) learn affine transformations to adjust the foreground position and reduce the geometric inconsistency between the source and the target images. While previous methods insert an object on an empty background image and focus on color harmonization, GCC-GAN (Chen and Kae 2019) introduces a deep learning model based on predicting color and geometric adjustment for replacing a given object with a new one in addition to inpainting missing empty regions. Finally, performing using copy pasting for image composition has been enhanced with refined mask prediction of the foreground as in (Arandjelović and Zisserman 2019).

Assessing the realism of generated composite images is a challenge. RealismCNN (Zhu et al. 2015) proposes a learning based approach to discriminate real images from composite ones by predicting a realism score while RGB-N (Zhou et al. 2018) introduces a two-stream Faster R-CNN network to detect the tampered regions given a manipulated image which we use in our study.

**Method**

Our approach relies on two key components; searching for suitable foregrounds to replace a selected one and performing image composition automatically. We first search for visually similar foregrounds from different classes, leading to placing objects in uncommon contexts. We then design an image composition model similar to the one proposed in GCC-GAN (Chen and Kae 2019), where we apply affine geometric and color transformations to the foreground before pasting it on the inpainted background. In the following, we assume we have access to a dataset of centered segmented objects with class annotations, otherwise, we can obtain it using an image segmentation algorithm.

**Foreground selection**

To find visually similar but semantically different foregrounds for a given query image, we search foregrounds of different semantic classes with the most similar features. Using local features allows us to have an object similarity with more emphasis on the shape similarity than using global pooled
features. We found that masking out the background of each object when computing local features leads to retrieving similar objects with similar masks, which is useful for visual blending through object replacement. We use the layer3 features of a ResNet-50 trained on the images from ImageNet (Deng et al. 2009) using MoCoV2 (Chen et al. 2020). To limit the memory footprint and computational cost, we reduce the dimension of each local feature from 1024 to 50 using Principal Component Analysis. Each local feature is $\ell_2$ normalized. Each foreground is then represented by a $14 \times 14 \times 50$ feature map. Given a query foreground object, we search the index and keep only the closest foreground from each class for our analysis as visualized in Fig. 2.

More formally, given a query image $I_q$ and the associated binary mask $M_q$, we select for each class $c$ the image $I_c$ and mask $M_c$ defined by:

$$ (I_c, M_c) = \arg \max_{(I,M) \in D_c} \langle f(I_q M_q), f(IM) \rangle $$

(1)

where $f$ is our feature extraction and $D_c$ the set of pairs of image and mask associated to class $c$. To enable fast online search, we build an index from pre-computed local features using the FAISS library (Johnson, Douze, and Jégou 2017) and search for similar foregrounds using the inner product between the flattened features.

In our analysis, we consider two ranking setups to select the pairs $(I_c, M_c)$ to use for our composition, based on the visual similarity of foregrounds as described above and on a distance between the different classes, both setups are explained in Fig. 2. For the first one, dubbed instance similarity, we rank the images according to their distance to the query, similar to equation 1 and we select the closest foreground in each class. For the second one, dubbed class similarity, we instead use the similarity of the average feature of each class $\frac{1}{|D_c|} \sum_{(I,M) \in D_c} f(I_q M_q)$, where $|D_c|$ is the number of images in $D_c$ with the average feature of the query class. While in the first setup we focus on the visual foreground similarity to rank the images, in the second one instead, we rank the closest objects according to the average similarity of
the class, introducing a notion of semantics in the similarity.

**Image composition**

Here, we assume we want to create a composite image using the foreground object of image $F$ associated to the mask $M_F$ and the background image $B$ excluding the object defined by the mask $M_B$. We consider a composition framework that predicts geometric and color corrections and applies them to the foreground object, similar to GCC-GAN (Chen and Kae 2019). We use affine transformations for both the spatial and color components. Particularly, we use spatial transformers (Jaderberg et al. 2015) as a differentiable module to spatially transform the foreground object, and denote $T$ and $C$ respectively the spatial and color transformations applied to the foreground. We denote by $g$ the network predicting the spatial and color transformation parameters $\theta_{ST}$ and $\theta_C$, it takes as input the concatenation of the masked foreground $FM_F$ and the masked background $B(1-M_B)$. We use the same architecture for $g$ as in (Lin et al. 2018). While, $T$ and $C$ are differentiable and have no trainable parameters, they take as input $\theta_{ST}$ and $\theta_C$. Fig. 3 illustrates our entire composition pipeline.

At test time, to compose our final image, we first compute the spatial and color transformation parameters $\theta_{ST}$ and $\theta_C$ using the network $g$ as shown in Eq. (2) to define a transformed foreground image $\hat{F}$ and a transformed foreground mask $\hat{M}_F$ as in Eq. (3).

$$\theta_{ST}, \theta_C = g(FM_F, B(1-M_B)) \quad \text{(2)}$$

$$\hat{F} = C(T(F, \theta_{ST}), \theta_C) \text{ and } \hat{M}_F = T(M_F, \theta_{ST}) \quad \text{(3)}$$

We then use the network InpaintNet from (Yu et al. 2019) to inpaint the background after removing the original object, thus obtaining $\hat{B} = \text{InpaintNet}(B(1-M_B), M_B)$. Finally, we compose the transformed foreground image and the background image into a final composite image:

$$\hat{F}\hat{M}_F + (1-\hat{M}_F)\hat{B}. \quad \text{(4)}$$

We train $g$ by creating synthetic examples as follows and as shown in Fig. 4: assuming we have access to segmented objects, we first extract an object and use its mask to create both foreground and background images; we then erode the border of both the foreground and background and jitter the foreground image using random affine color and spatial transformations obtained from a normal distribution $\mathcal{N}(0, 0.1)$ as perturbations from the identity of each transform. We use two different losses for the spatial and color transformation parameters $\ell_2$ distance between the predicted and target parameters to undo the spatial perturbation. For the color, such an error was not representative of the visual similarity between the transformed images. We use instead the $\ell_1$ distance between the original foreground and the corrected one using the predicted color transformation. Note that mask erosion of the foreground and background is important to remove obvious visual clues and to make the training more challenging.

Figure 4: Examples of color and spatial random transformation used in our synthetic dataset to train our model. From left to right, we show the original image, the jittered one, and the overlap of eroded foreground and background that we input as 6 channels. The top row shows color modification associated with mask erosion. While the bottom one shows an example with spatial pipeline jittering.

Note that our training and composition procedure are much simpler and more stable than the one proposed in (Chen and Kae 2019), which uses multiple adversarial losses and that we were unable to reproduce.

**Dataset**

In order to demonstrate our search and composition method, we use OpenImages dataset (Kuznetsova et al. 2020), a large collection of objects from diverse annotated classes with their mask annotations. We subsample a set of relevant segments by filtering out small objects (< 64 × 64 pixels) and images of low quality computed using the image quality estimation network Koncept-312 (Hsu et al. 2020) leading to a dataset of 37 233 images from 319 object classes. Note that the quality of the obtained compositions heavily depends on the diversity of annotations in the dataset, and that using a larger image set would definitely lead to better image compositions.

**Results**

In this section, we demonstrate the performance of our composition method by highlighting the importance of using spatial and color corrections through comparison with baselines both using a tampering detection metric and visually. We then present the human study that we perform to compare the two class sorting methods for selecting foregrounds of our composites, and show images that obtained unanimous human ratings.

**Composition baselines**

We consider three baselines for our composition algorithm. The first one is based on simple object copy pasting, enhanced with inpainting the region of the removed object. The two other baselines are based on Poisson image blending (Pérez, Gangnet, and Blake 2003). While this algorithm is designed for inserting a foreground
object on a background image, we adapt it with an inpainting step to fill in the removed initial object mask, we name this baseline “Poisson”. In the last enhanced baseline that we name “ST+Poisson”, we apply our learned spatial transformation module to adjust the foreground spatially, and process it using Poisson blending.

**Quantitative evaluation** RGB-N score is a tampering detection score presented in (Zhou et al. 2018), it represents how realistic an image is by detecting tampered regions and averaging their detection scores. In Table 1, we report this score averaged over 1000 images sampled from the top-10 compositions obtained with our two foreground ranking strategies and the different baselines presented above. While copy pasting composites are systematically detected as tampered ones, our composition method obtains lower RGB-N score than all baselines both for top-10 compositions obtained with class similarity or using instance similarity. Also, we note the very clear boost given by our spatial transformation both with our composition method and the Poisson composition baselines.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class sim.</th>
<th>Instance sim.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real images</td>
<td>59.24</td>
<td></td>
</tr>
<tr>
<td>Copy-paste</td>
<td>97.49</td>
<td>97.45</td>
</tr>
<tr>
<td>Poisson</td>
<td>73.06</td>
<td>72.55</td>
</tr>
<tr>
<td>ST+Poisson</td>
<td>65.42</td>
<td>64.02</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>58.01</strong></td>
<td><strong>56.95</strong></td>
</tr>
</tbody>
</table>

Table 1: Tampering RGB-N scores for real and composite images computed over 1000 samples. (lower is better)

**Qualitative comparison to composition baselines** In Fig. 5, we show a comparison of our composition algorithm with the baselines including a simple object copy paste. Our model is trained to undo synthetic affine color and spatial transformations, therefore, it predicts suitable geometric and color transformations to adjust the spatial arrangement of the foreground object and harmonize its appearance in the background image. On the contrary, the Poisson blending baseline suffers from color bleeding and is unable to resize and place the foreground object.

**Human study** We design an experiment where human raters are asked to evaluate different compositions obtained from the same original image. The goal is to understand how real, surprising and liked our compositions are given the class selection strategy for the new foreground. We thus rank the candidate classes either using our instance similarity or our class similarity strategy. For each annotation task, we sample four composite images from four groups defined by the rank of the selected composition (between 1 and 5, 6 and 10, 11 and 20 or above 20). For each of the class selection strategies, we randomly sample 200 tasks obtained from the same original images, and each task is presented to 5 different raters, leading to 1000 task evaluations per class selection strategy. Raters are shown the original image and four shuffled compositions and asked to select the most surprising composition, the one they like the most and the most realistic one independently.

In Fig. 6, we compare the ratings obtained by each group and for each search method; using class similarity or instance similarity to rank the selected foregrounds. We observe a much clearer correlation of the surprise and realism ratings with the rank groups from the class similarity selection - smaller ranks corresponding to more realistic and less surprising compositions - while little correlations are observed with the instance similarity. The observed correlations for class similarity are significant, as checked using a Pearson’s Chi-squared test with p-values (0.002 for likeability, < 0.001 for surprise and < 0.001 for realism). Instead, using instance similarity foreground ranking method, only the correlation...
with realism is significant with a p-value of 0.002, the other p-values being larger than 0.05. We show examples of images with unanimous ratings in Fig. 7. We observe in these examples that composites generated by our method can be very realistic, by replacing foreground objects in similar contexts (birds or animals replacements). In contrast, when the foreground class is picked far from the original one, the context may be very different, resulting in surprising results (e.g. giraffe in the city, crocodile in plate). The most liked compositions, more difficult to analyze, can be explained in some cases by a judgment of the image aesthetic, or preference for some object class.

Human ratings (sorting with instance similarity)

Figure 6: Human study: comparing realism, likeability, and surprise ratings for compositions obtained with class or instance similarity ranking. We represent the proportion of each group in being selected as real, liked or surprising on 1000 tasks.

Conclusion

We presented a new image composition approach for creating uncommon object combinations based on visual similarities, designed to help artists search and visualize compositions interactively. Our approach simplifies image composition by using a geometric and color prediction network trained on synthetic data in combination with a state-of-the-art inpainting model. There is a great potential in using our composition approach as a data augmentation method for improving instance segmentation and image tampering detection. Our human study shows that we can control the realism and surprise by considering class similarity instead of foreground similarity alone. In future work, our approach could benefit from using a larger set of objects with mask annotations, or searching images for non-annotated objects. Finally, we believe that using recent image generation models conditioned on natural language could be a great advance in visual blend generation.

References


Figure 7: Image selected through our human study with most liked, most surprising or highest realism ratings. By pairs, left columns represents the original image and the right columns our composite image. Images that we show have at least 4 unanimous ratings among 5 raters.


Xiao, P.; Linkola, S. M.; et al. 2015. Vismantic: Meaning-making with images. In *ICCC.*


**Image attributions** All images used are CC-BY licensed https://creativecommons.org/licenses/by/2.0/. We provide below the landing url for each of them. Images were either cropped to center on a given segment, or modified by replacing that segment with another one. We thank the authors for sharing their images.

---

Jon’s pics https://www.flickr.com/photos/nakedcharlton/2467975539
Elena Roussakis https://www.flickr.com/photos/mom2zofia/20544855741
Graham Richardson https://www.flickr.com/photos/didbygraham/8097017171
Funk Dooby https://www.flickr.com/photos/funkdooby/9266499179
NASA Remix Man https://www.flickr.com/photos/remix-man/3944371937
Boris Kasimov https://www.flickr.com/photos/kasia69/1163561934
Hernan Gonzalez born1945 https://www.flickr.com/photos/125677130/n00/5198266561
Sharron https://www.flickr.com/photos/photos/lownote/5963290835
Purple Slog https://www.flickr.com/photos/purpleslog/2895239551
Symmetry https://www.flickr.com/photos/hsymmetry/15583330630
Sarah Ackerman https://www.flickr.com/photos/sackerman519/14145216183
Alex Brown https://www.flickr.com/photos/alexbrn/4780066832/
Mike Shelby https://www.flickr.com/photos/mikeshelby/7756265238
Bulaclac Parapara https://www.flickr.com/photos/bowakon/2598439551
Geoffrey Fairchild https://www.flickr.com/photos/gefairchit/4283659992
michimaya https://www.flickr.com/photos/michimaya/4898288174
storebakkebruse https://www.flickr.com/photos/tusselka/16525955992
Jessie Pearl https://www.flickr.com/photos/erwillinger/11148037996
Mike Baird https://www.flickr.com/photos/mikebaird/2114440009
Lisa Cyr https://www.flickr.com/photos/photos/borderleys/4099272800
Kim Aldström https://www.flickr.com/photos/photos/kimatra/3051613191
Ron Knight https://www.flickr.com/photos/photos/sussexbrider/8082808115
wes_arnenberg https://www.flickr.com/photos/photos/western/2413035236
evan_p.cordes https://www.flickr.com/photos/photos/peechy/2111809234
piontmammam https://www.flickr.com/photos/photos/mamnamme/16032215481
diane cardell https://www.flickr.com/photos/photos/duncardell/16757242653
Donald Hobern https://www.flickr.com/photos/photos/dhobuer/14551965235
Derek Keats https://www.flickr.com/photos/photos/dkeats/12223031243
aseiff https://www.flickr.com/photos/photos/aseiff/3498048272
Roxanne King https://www.flickr.com/photos/photos/roxyeking/8364971012
Britt Reints https://www.flickr.com/photos/photos/brittreints/8671602383
U.S. Department of Agriculture https://www.flickr.com/photos/photos/usdagov/6276665873
Bernt Rostad https://www.flickr.com/photos/photos/brestad/149090930439
ppc1337 https://www.flickr.com/photos/photos/wh/14151258744
jenn.b https://www.flickr.com/photos/photos/147146050/n05/4532625630
Biev Sykes https://www.flickr.com/photos/photos/bisykes/2146841578
mauroguanandi https://www.flickr.com/photos/photos/mauroguanand/4445312169
Du Truong https://www.flickr.com/photos/photos/duan/130448072/n02/16776905479/
Donald Hobern https://www.flickr.com/photos/photos/dhobuer/5079221577
Parker Knight https://www.flickr.com/photos/photos/rocketboom/6901703403
Keith Cooper https://www.flickr.com/photos/photos/cooperweb/5861407488
Shardayy https://www.flickr.com/photos/photos/shardayy/5728844632/
Kim Harston https://www.flickr.com/photos/photos/kimharr/7106129340865
HackBiz https://www.flickr.com/photos/photos/mg/6315780494
Creative Tools https://www.flickr.com/photos/photos/creative_tools/16680258211
cherryblossom in japan https://www.flickr.com/photos/photos/cherryblossom_in_japan/382999121
Christian Heilmann https://www.flickr.com/photos/photos/codepo8/6739104495
Paul Hudson https://www.flickr.com/photos/photos/pahudson/9922320853
Aaron Vowels https://www.flickr.com/photos/photos/979643640/n00/52131091
*paras* https://www.flickr.com/photos/photos/parasch-blechler/132527108
la riviere https://www.flickr.com/photos/photos/lairi/557304866
Ben Kucinski https://www.flickr.com/photos/photos/benku/1419761333
A Davey https://www.flickr.com/photos/photos/adavey/14122710960
Pavel Rybin https://www.flickr.com/photos/photos/pavelrybin/164426500/
A Deep Learning Pipeline for the Synthesis of Graphic Novels

Thomas Melistas,1* Yannis Siglidis,2* Fivos Kalogiannis,1* Ilan Manouach3*
1School of Electrical and Computer Engineering, National Technical University of Athens, Greece
2Department of Mathematics, ENS Paris Saclay, France
3Department of Media, University of Aalto, Finland
melistas.th@gmail.com, ioannis.siglidis@ens-paris-saclay.fr, phoekealogiannis@gmail.com, ilan.manouach@aalto.fi

Abstract
In this paper, we present what is to the best of our knowledge, the first deep learning pipeline to produce a synthetic graphic novel. Our method can synthesize from scratch engaging sequences of graphic novel pages, focusing on the Manga genre. To achieve this, we extract images and text from around 670 thousand Manga pages, which we use separately in order to train state-of-the-art generative architectures, such as GPT-2 for text generation and StyleGAN2 for image synthesis. Using these as sources of synthetic content, we develop a set of algorithmic aesthetic rules in order to bring together complete and continuous Manga pages.

Introduction
There is little consensus among comics scholars on whether comics is a language, but it’s relatively agreed that comics is a sequential system of communication, consisting of both linguistic and non-linguistic signs (Groensteen et al. 2007). Comics create, most commonly, a narration, the contents of which are images and text, while its form is the panel layout, the placement and shape of text bubbles and the succession of panels and pages. More generally, we can consider comics synthesis as the generation of images and text, as well as their common arrangement, in a way which suggests a sense of narration and/or dialogue. While comics traditionally unfold a structured storyline and contain text and images that are directly related, types of comics that explore more unconstrained and creative directions have emerged. Our approach is motivated by such works, since we produce images that are directly related, types of comics that explore

work is mainly positioned inside this context, aspiring to question and challenge creativity in the multi-modal and complex setting of graphic novels.

We focus on a specific graphic novel form which originated in Japan, called Manga. Manga comics come in a huge variety and quantity, being classified into many genres on the basis of their targeted audience, the main subject of their plot and their artistic style. They were originally distributed in black-and-white prints. Nowadays, vast web-communities of Manga enthusiasts, known as scanlators, share scanned Manga comics in low quality through the web, usually poorly translated in English. Both because of their world-wide popularity and their abundance, but also because of their automated production process1, they are a form of art for which we could speculate their, at least partial, automation through Artificial Intelligence in the coming years. To support even more this claim, Manga follow a certain consistency in drawing style, which differentiates them a lot from other types of comics, and are found in abundance, with many successful Manga series consisting of thousands of pages.

We present our contribution, a deep learning pipeline for synthesizing complete and continuous pages in the form of a graphic novel. Our work consists of two main parts: (a) the necessary pre-processing or content extraction step, namely the extraction of images and text from raw Manga comics and (b) the synthesis of the content and its assembly into the form of a graphic novel. We thoroughly describe all steps of the above process and underline the challenges we encountered, as well as the techniques we adopted to surpass them.

Firstly, we describe the pre-processing procedure. The first step is the extraction of Regions of Interest from Manga pages, namely image panels and text bubbles, using a region proposal based convolutional network. Next, we train a U-NET segmentation network (Ronneberger, Fischer, and Brox 2015) to precisely segment and isolate text bubbles, which are then inpainted in order to obtain clean images. Finally, we get text transcriptions of the detected areas, using

1A typical production line of manga comics for example involves dozens of people handling specialized roles in a quasi-taylorist production belt, often in ways that have been criticized for resembling a sweatshop, while distribution has been increasingly involving massively digitized operations of logistics and global supply chains. Comics is, an industrial form of artistic expression.
an Optical Character Recognition model, which we domain-adapt by fine-tuning it to commonly used Manga fonts, increasing its recognition performance. For all these steps, we have used all resources that were available to us, namely existing datasets, manual data annotation and ad-hoc synthetic datasets.

For the second step, we start with training a StyleGAN-2 architecture (Karras et al. 2020) on the inpainted Manga images that we previously extracted. To alleviate the low quality and diversity of the generated samples, we explore two different approaches: (1) we train a conditional model, providing labels that we acquire from a tag estimator trained on Anime art and (2) we perform transfer learning using a model pre-trained on Anime faces. To fit the industry standards for image quality, we chain the generation procedure with a super-resolution up-scaling network trained on Manga content. Next, regarding text generation, we fine-tune a GPT-2 language model (Radford et al. 2019) on text extracted from our Manga dataset, as well as on a diverse set of monolingual corpora from different genres of literature. We finally generate both image and text content in a sequential manner and place them inside randomized and standard panel layouts, bringing them together in the form of graphic novel pages.

Related Work

In this section, we discuss relevant research, regarding the comics medium and the procedure of content synthesis. We can make a first distinction between: (a) research that focuses on the analysis of comics and extracts information that is crucial for specific tasks and (b) work that deals with the synthesis of Manga related content, such as animated characters.

The first approaches on the field of comics analysis, focus on using traditional computer vision techniques for the extraction of basic comics features. One of the earliest tasks, providing motivation in this research field, was panel and text extraction, mostly oriented towards automating the process of formatting comics for reading in mobile devices (Yamada et al. 2004; Ho, Burie, and Ogier 2012; Li et al. 2014) or copyright protection (Sun and Kise 2013). With the rise of deep learning, research steered, with great success, towards the use of neural networks for similar or other, previously unexplored tasks. The creation of appropriate datasets, such as eBDtheque (Guérin et al. 2013) and Manga109 (Fujimoto et al. 2016) (both of which we also use in our work) helped to build more robust tools for detecting and extracting comics features, such as panels (Ogawa et al. 2018; Zhou et al. 2020), text bubbles (Dubray and Laubrock 2019) or characters (Qin et al. 2017), using deep convolutional architectures. Their extracted contents can be used in many ways, for example to create an indexing system with content analysis (Nguyen, Rigaud, and Burie 2018), sketch-based Manga retrieval (Matsui et al. 2016), semi-automatic comic colorization (Furusawa et al. 2017) or making comics more accessible to the visually impaired (Rayar, Oriola, and Jouffrais 2020).

Regarding the synthesis of comics, little work has been done towards the form of narration itself, with a few notable examples, such as the synthesis of Manga-resembling layouts (Cao, Chan, and Lau 2012). Previous work has mainly focused on synthesizing Manga related artwork. The evolution of the GAN architecture during the past years, has increased the interest for generation non-photographic animated characters, typically found in Manga, although it is most commonly used for generating photographic images. In (Su et al. 2020) a GAN architecture is trained to create Manga faces from photographs, while preserving the original face features. A notable example of image generation is the work of Gwern, “this waifu does not exist” (Gwern 2019b), a website hosting a StyleGAN-2 generator for female Manga character portraits with some text, independently generated by GPT-3 (Brown et al. 2020), accompanying each. The basic image generation component of this work was recently updated, introducing a revised StyleGAN-2 architecture, and was featured in the “this anime does not exist” project (Aydao 2021). Important changes of Aydao’s approach are the doubling of the feed-forward embedding layer’s width (consequently doubling the dimension of the latent vector) and decreasing the amount of regularization, leading to slower but more stable training.

Content Extraction

Throughout our work, we use a custom private dataset, consisting of 667,181 black-and-white Manga pages with English text in 72 dpi. Unfortunately, we are not in the position of making this dataset public or publishing the extracted content, as it is part of a private collection assembled from diverse sources, subject to copyright law. Furthermore, we do not perform any human annotation on the above dataset, so the evaluation of all the techniques we present next is mainly done visually from random samples, since our data is unlabeled.

Panel and Text Bubble Detection

As a first step we detect panels (areas in which images are located) and text bubbles (areas in which text is located). As a primary annotated resource for this task we use the Manga109 dataset, which contains handcrafted annotations for panels, characters and text bubbles in the form of rectangular bounding boxes. We found, that training a Faster R-CNN model (Ren et al. 2015) on this small dataset is effective for extracting bounding boxes on our larger dataset. Faster R-CNN incorporates a Regional Proposal Network that shares features with a detection network and is widely used in relevant tasks, even for extracting comics features. Our implementation is largely based on the MMDetection framework (Chen et al. 2019).

It should be noted, that while it is common for Manga panels to have boundaries that are not parallel to the borders or are more complex than quadrilateral, there is no large annotated dataset which contains non-rectangle polygons or masks for object detection. Moreover, modeling rectangular (or even square) images of a fixed resolution is the predominant approach, used by most of the existing computer vision and image generation architectures. Adding white margins
to the rectangular images is a possible solution, but while it can increase expressiveness and data variability, it introduces white areas that dominate the generated images and complicates the final page assembly. As we mention below, we arrive to the solution of cropping and resizing accordingly.

Finally, we develop an algorithm that sorts the detected panels and text bubbles according to the Manga reading order, while it associates each bubble with the panel it belongs to. Manga comics are read from right to left and from top to bottom. This applies to both panels and text bubbles. We consider a panel preceding another if it is located higher on the page or if it is on the same level but on the right-hand side. Panels are considered to be on the same level if the horizontal border lines of one are contained on those of the other or if the difference between their respective upper or lower borders is smaller than half the height of the shortest panel. Each bubble is then associated with one panel, based on the distance of their centers. Bubbles that belong to the same panel are sorted following the same procedure. The above algorithm is robust to unconventional panel layouts, common in Manga comics and enables us to extract and encode structural information, such as the sequences of images and the text-image correspondence, as well as getting a complete and ordered story from the extracted text.

Text Bubble Segmentation

In this step, we remove all the text bubbles from the extracted images, as such a visual feature would insert noise and dominate the generated samples. To achieve this, we first detect the exact region the bubbles occupy and then use proprietary software to do content-aware inpainting of the area underneath. The Faster R-CNN model, that we have used above, is not sufficient for this task, as it only can provide us with a rectangle bounding box. An effective solution to this problem would require pixel-level masks which cover the exact area that needs to be inpainted. This is rather an image segmentation task, which we approach using a U-Net architecture (Ronneberger, Fischer, and Brox 2015), following the implementation of the fastai library (Howard and Gugger 2020).

We use the labeled eBDtheque dataset (Guérin et al. 2013) to train our model. It contains pixel level masks for text bubbles of comics that span various styles and traditions. Since it is a small dataset, consisting of only 100 annotated comic pages, we augment it using custom-made synthetic data. To produce them, we place handcrafted text bubbles, with fill-in real text, on pages whose bubbles we have removed successfully during a previous iteration. To achieve a first rough estimation of the bubble area, we have implemented a flood-filling algorithm to find connected components based on pixel intensity around letter markers, extracted by the Faster R-CNN model that we have previously trained. To make our model more robust, we train it on augmented versions of both the original and our synthetic dataset. Images are randomly cropped and rescaled, followed by a random affine transform, to which random brightness is applied subsequently.

Text Transcription

The last step of this process is the extraction of text from the detected text bubbles. To achieve this, we use the Tesseract Optical Character Recognition (OCR) Engine (Smith 2007). Using a model that was pre-trained on English text led to character recognition of poor performance. This is not unexpected, since comics and especially Manga contain specific and uncommon fonts, as well as not casual monochromatic backgrounds and disrupted or skewed text, which the initial trained model is naturally unaware of. Unfortunately, we could not use the Manga109 dataset for this purpose, as it contains text transcriptions only in Japanese, and as far as we know, no other annotated dataset was available.

To produce them, we place handcrafted text bubbles, with fonts resembling those found in Manga comics. As a text corpus we have used “Ulysses” by James Joyce, because of its casual tone and plethora of neologisms and onomatopoeia. The text was partitioned into chunks of varying sizes and then split into lines of varying lengths, in order to better represent text bubbles. Specifically, the number of words contained in each bubble was chosen uniformly in the range of 1 to 20 and line breaks were added to split each bubble to approximately 4 lines, with some added randomness. Also, some punctuation commonly used in Manga, such as triple dots and exclamation marks, were manually added. Using 40 fonts, common in Manga, we generate a total of 20,000 text images with varying word and line count, different backgrounds, font sizes, orientation, blurring and skewing. Some examples can be seen in Figure 1. We find that the fine-tuned model significantly improves the quality of text transcription.
Synthesizing a Graphic Novel

In this section we discuss the method we adopt in order to generate synthetic content, as well as the algorithmic procedure that we follow in order to assemble a graphic novel.

Synthesizing Images

We consider the development of the GAN architecture to be a pivotal moment in synthetic content creation, which at first proved its remarkable ability to generate mono-categorical photo-realistic images, such as human faces. One of the most influential architectures, widely used in a wide variety of creative applications is StyleGAN2 (Karras et al. 2020), yielding state-of-the-art results in generative image modeling. Its advantages among alternatives include its ability to be trained on images of higher resolution and the feasible computing resources needed compared to other methods, which has led to its adoption by a community of StyleGAN artists.

After the aforementioned extraction and the separation of images with aspect ratios close to 1x1, we end up with 1.7 million monochromatic images, which are resized and cropped to fit the 512x512 resolution. Training a StyleGAN2 architecture from scratch to this data proved insufficient, resulting in non-convergence. This was either reflected on a high Frechet Inception Distance (Heusel et al. 2018) measure (more than 30) or on complete divergence. Just tuning the hyper-parameters (for example decreasing the regularization or the learning rate) did not solve this issue. We attribute this poor training performance both to the absence of a center in our data, as well as to their poly-categorical nature and to their complex textures, something which is not the case in most traditional datasets which have been used for evaluation purposes, such as FFHQ (Karras, Laine, and Aila 2019). To overcome this, we experiment with two standard approaches: (a) boosting the learning process with label conditioning and (b) transfer learning.

The first approach requires a meaningful categorization or labeling of our images. The idea that organizing the diversity of our dataset could result in improved performance was explored in (Oeldorff and Spanakis 2019), which showed that meaningful conditions enable the model to learn a larger number of modes and produce more detailed, diverse and controllable outputs. Additionally, it seems that although simple label conditioning is supported in the standard implementation of StyleGAN2, it is largely unexplored by the (art) community.

To label our images, we utilize a multi-tag ResNet classifier (He et al. 2015), pre-trained on the 5543 most popular descriptive tags of the Danbooru dataset (Anonymous, community, and Bransen 2021). Danbooru is currently the largest available public dataset of anime-style images, commonly found in Manga. Each image is accompanied by around 30 tags coming from a total of 43k predefined tags. Unfortunately, a single annotation by itself is rarely descriptive of a single image. Thus, we predict the tag scores for all our images and apply incremental PCA (Ross et al. 2008) to reduce their dimensionality to 20. Next, we perform clustering, using the incremental k-means algorithm (Pham, Dimov, and Nguyen 2004) in order to extract 20 categories in total. By adding label information, we obtain more diverse content with a lower FID, but which is much more figurative. A few examples can be seen in Figure 2. Although we did not try other variants of this procedure, we strongly suggest it as a subject of future work.

For the second approach, we fine-tune a publicly available model that has been trained on 512x512 female Manga faces (Gwern 2019a) on our dataset. This model is the core of the aforementioned “this waifu does not exist”. We observe that by following this approach, we manage to model face characteristics with a higher fidelity, which can in fact help to make our comic more engaging. On the other hand, this approach exhibits less diversity, and as expected is much more biased towards generating female faces. Some examples that showcase this can be found in Figure 3.

Sampling Procedure

We sample the latent space of our model using a 513-dimensional vector (512 for the noise vector, plus one for truncation $\psi$) of an interpolated time-series of stock volumes. This results in a continuous sequence of transformations, a procedure that is generally used for the exploration of the latent space and which creates a sense of action for various characters that appear locally for certain latent vector values. After generating about 130K images, we manually classify them using the active learning technique of Pool-Based sampling (using as informativeness the maximum probability of any category), to four ad-hoc visual categories, plus a fifth which is used to discard images with ambiguous content. As the produced images have a 512x512 resolution, we bring them to the industry standard of 1024, using the domain-specific up-scaling model “waifu2x” (Nagadomi 2018).

Synthesizing Text

Language models assign probabilities to sequences of words and are commonly used to generate text, by iteratively choosing a word given the previous context. GPT-2 (Rafford et al. 2019) was until recently the state-of-the-art model for language generation, superseded by its successor (Brown et al. 2020). It consists of billions of parameters and it has been trained on a huge corpus of texts scraped from the web. Even from early experiments, we have noticed its unique ability of domain adaptation when fine-tuning to a certain author or genre; its results could be seen in broad terms as the impersonation of an author or the imitation of the genre.

In our work, we use a distilled version of this model (distilGPT-2), which is contained in the Hugging Face library (Wolf et al. 2020). Knowledge distillation (Hinton, Vinyals, and Dean 2015) is a technique for reducing the size of a model, while decreasing its accuracy by a small factor. The distilled GPT-2 model is twice as fast and consists of 37% fewer parameters than the smallest full model available. We notice that by using the original versions of GPT-2, the sampled outputs can be a bit more poetic and fruitful, but much less reliable in coherence and syntactic precision. We train this language model on the Manga text that we have extracted above, where we use special tokens to separate text that belongs to different bubbles and panels.

To introduce more interesting and diverse textual content,
we train language models on a wide variety of publicly available monolingual corpora of literature. For this purpose, we compile separate English corpora from various works of Poetry, True-Crime, Science Fiction and Buddhist literature. Models which are fine-tuned on each of those different language models will be used later as distinct “voices” (Poetry will play the role of narration), to give the impression of a dialogue which unfolds between distinct characters, focusing on different subjects and emotions. To make the output more appealing and to exclude writing styles unrelated to graphic novels, we only keep sampled sequences that follow a set of simple semantic rules (for example ignoring chapter names or references), without a great loss of continuity, as texts of similar styles are generated in sequence. We observe that these models produce more rich and engaging samples than the one which was trained on the text extracted from our Manga dataset.

**Assembling a Graphic Novel**

As the final part of the described pipeline, we investigate the assembly of a comic from two independent and locally coherent streams of image and text content. We experiment with two processes to generate panel layouts for a fixed page size of 2480x3508px: (a) a randomized approach of generating panels in multiple scales and rectangle shapes and (b) a standard 5 row by 4 column layout of square images.

We first develop a randomized layout synthesis, that follows several fixed constraints. We design a recursive function, which starts from a higher level and based on a biased coin-flip, decides whether it should split in half the current panel space, either horizontally or vertically. Each time, the level is decreased by one, starting from 4 and it stops when it reaches the number zero (see Figure 4). Additionally, while splitting a level in half, it also decides whether the split will be parallel to the borders of the page, or whether it should include a certain amount of randomly selected small inclination. Moreover, we randomly decide if the borders of each panel will have a gutter (the value of which is again selected randomly) or if they will be a so called full-bleed. All those distributions are parametrized at panel level and their detailed parameters are estimated through experimentation to fit certain aesthetic criteria. Figure 5 contains a few examples of different layouts that are synthesized using this procedure. The non-randomized approach (b) is not discussed because of its simplicity. It is mainly introduced, to balance out the fuzzy elements of the synthetic text and image content and to allow the reader to effectively engage with the current version of this work.

For each of the above procedures we have used different ways for selecting synthetic content. In the case of (a) we follow again a manually tuned random sampling procedure for each level, were we select images based on their categories. Moreover, to avoid having images with a lot of variation in small levels we use a very simple measure of Shannon-Entropy. To fit an image inside each panel, we resize each image to the smallest dimension of the panel, while keeping its aspect ratio constant and we crop it with equal margins from the borders of its largest dimension. For the more coherent version of the squared layout we ignore the categorical labels and use 4 local images which come from
As a next step we allocate a page and a panel with a custom amount of bubbles for each level, which we will then fill with text, both in the form of dialogue and of narration. For the purpose of dialogue we use three different types of text bubbles, which would represent the three different voices of our previously fine-tuned language models and a (final) fourth one which plays the role of narration. In the case (a) of randomized panels, we allow at most one voice-over per panel and up to four bubbles, the amount of which is randomly selected from a range of values related to their level. In the case of (b), we evenly and randomly distribute the narration text boxes and the bubbles of a given number inside the page. Also, in both cases, we bound the total amount of generated text per level, also taking into account its function - as narration or as dialogue. We implement a recursive function to best fit our text inside a given rectangle area and then re-scale our bubbles, so that their inscribed rectangle would properly fit our text. To locate the area, inside which we would place those bubbles, we construct a Boolean mask which we incrementally update, starting by adding the narration and afterwards by placing each dialogue bubble, while adding a constraint to the total area which can be covered by a bubble inside the panel or inside the page (in cases (a) and (b) respectively).

In terms of content, we use a poetry language model for the purpose of narration, as it provides a sense of poetic continuity throughout the pages. Moreover, for the dialogue bubbles we use three different types of language models which we associate with different qualities and emotions. First we have a Buddhist model, that with a more spiritual, zen and wise vibe. Then we have a Sci-Fi model that is futuristic, exhibiting a certain amount of complexity in situations and techniques. Finally, a True-Crime literature model that creates a sense of action and suspense is included. To conclude, a visual example of method (a) can be seen on Figure 6 and one of method (b) on Figure 7, which was the one that made it to the "final cut".

5 sequences per page in the given order. This plays the role of action or character unfolding.

In Figure 4: Panels and their associated levels.

Figure 3: Progress of transfer learning from a model pre-trained on Manga female faces @ kimg=1490.
**Future work**

In this work, we manage to construct a pipeline which synthesizes pages that resemble the style of Manga comics. The generation of image and text content, however, has been done independently and their assembly has been aesthetically pleasant and engaging but not necessarily indicative of our dataset. Associating images with text in a meaningful way and arranging them in a semantically consistent order is a critical next step, that will allow us to research creative narration in the multi-modal setting. Relevant tasks have been well researched, such as image captioning (Herdade et al. 2020) or image generation conditioned on text descriptions (Ramesh et al. 2021). However, it can be argued that the text and image interrelation, in the setting of comics, is not as straightforward and strict as that of an image and its caption.

An architecture that emerged recently and stirred up interest regarding the connection of visual representations to natural language is CLIP (Radford et al. 2021). To sum up its basic functionality, CLIP learns joint text and image embeddings, that allow the computation of a (cosine similarity based) matching score between images and text. We believe that analogous approaches should definitely be explored for the task of end-to-end comics synthesis. Ideally, a graphic novel generator is a multi-modal system that learns from and generates parallel streams of image and text, structured accordingly. As we previously mentioned, the function of images and their relation to text in graphic novels is unique and unexplored. Image and text are equally important parts of the same narrative, with no strict hierarchical relationship between them being implied. Therefore, simply generating one modality conditioned on the other is an approach limiting to the comics medium. For example, images can serve the purpose of capturing the reader’s attention for the text to find its meaning, while in other cases text can simply include a non-important dialogue as what is happening in the image is more important for the story.

Thus, in order to fully model a graphic novel, one could model narrative as a latent and discrete time series underlying both the image and text data, from which new images and text can emerge. This work does not claim to have tackled this problem, but rather it is the first applied and complete study towards this direction, using contemporary tools and suggesting their limitations.

**Discussion**

The comics industry has been quite reticent in embracing the complex nature of technological developments in artificial intelligence. But this situation might soon change. The online abundance of digitized media content, available through third-party groups of comics fans, the increasing convenience of programming language frameworks and machine learning libraries, the secularization of knowledge through e-learning and the plummeting prices in specialized hardware might contribute to reach a critical point where artifi-
Sational intelligence will be gradually integrated in the comics pipeline. Synthetic and generative processes might soon reshape the ways we produce, consume, archive and distribute comics artifacts. A more wide adoption of artificial intelligence in different strata of the industry might reconfigure existing readership(s) market(s).

Our research aims to explore the conditions for synthesizing graphic narratives and comics with the use of deep neural networks. This may result to a better understanding of creativity for comics artists and cartoonists but might also contribute to multiple applications of multi-modal expressive communication that has become our primary modality in sharing and shaping representation of our worlds. Modeling a graphic narrative still remains a challenging task, which has generally been ignored by the deep learning and computer science community. Researching and understanding this multi-modal, discrete and symbolic procedure involved in the production of comics, provides a very challenging task which can unite comics artists and deep learning engineers and potentially augment human creativity in ways which have never been experienced before. Finally, we suggest that reverse-engineering or modeling parts of this procedure can even provide us with mathematical and technological tools, which could profit other fields unrelated to artistic practices.

Acknowledgments

Ilan Manouach is the founder of Applied Memetic, a synthetic media company committed to explore the impact of deep learning in the comics industry. This research and the development of its early AI models has been materialized using the company’s resources.
References


Gwern. 2019b. This waifu does not exist. https://www.gwern.net/TWDNE.


Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network.


A Case-Based Approach to Creating Movie Poster Compositions

Jorge Alejandro Ramírez Gallardo  
Mechatronic and Industrial Engineering  
Instituto Tecnológico Autónomo de México (ITAM)  
Mexico City, Mexico  
jorge.ramirez.gallardo@itam.mx

Rodrigo Castillo Gómez  
Computer and Mechatronic Engineering  
Instituto Tecnológico Autónomo de México (ITAM)  
Mexico City, Mexico  
rodrigocastillo97@gmail.com

Andrés Gómez de Silva Garza  
Department of Computer Engineering  
Instituto Tecnológico Autónomo de México (ITAM)  
Mexico City, Mexico  
agomez@itam.mx

Abstract

This paper describes a system for the design or redesign of movie posters. The main reasoning strategy used by the system is case-based reasoning (Leake 1996), which requires two important sub-tasks to be performed: case retrieval and case adaptation. We have used the random forest algorithm to implement the case retrieval subtask, as it allows the system to formulate a generalization of the features of all the top matching cases that are determined to be most relevant to the new (re)design desired. We have used heuristic rules to implement the case adaptation task which results in generating the suggestion for poster composition. These heuristic rules are based on the Gestalt theory of composition (Arnheim 1974) and the requirements specified by the user.

Introduction

The main objective of our system is to design new posters for movies based on certain desired characteristics. On the other hand, the system can also be used to redesign an already existing poster for a movie. The system’s name is CACARO, which stands for CAse Composition Algorithm for the (Re)design of One-Sheets. The term "one-sheet" is used in the movie industry to refer to a movie poster. The generic name "Cácaro" is used in Mexico to refer to the projectionist in a movie theater—anyone who is a projectionist is called "Cácaro" by the paying public.

The user describes a new problem to the system by specifying the most important desired characteristics and CACARO fills out the description of the poster by proposing additional characteristics. These additional characteristics are decided based on already known (previously existing) posters within the system’s case base (which contains 103 cases).

In order to transfer the knowledge from past cases to the new situation, the abstraction and generation of knowledge required to describe the new poster is based on the Gestalt theory of composition.

The structure of the rest of this paper is as follows. First, we provide a section in which we discuss the theory of composition in the context of poster composition design. Then we present sections on decision trees and random forests, including how we use them for case retrieval in our system, and on heuristic rules and how we use them for case adaptation in our system. We then include a section that provides implementation details. Additionally, we have a section in which we fully trace, and describe the results, of running one test problem on our system and include a brief discussion of the results from two additional test problems. Finally, we provide a section which includes discussions, conclusions, and future work.

Theory of Composition

To propose the composition of a movie poster, CACARO uses Gestalt theory and its principles of perceptual organization. The main approach of Gestalt principles to art is based on Arnheim’s vision (1974), which recognizes that the whole (in this case, the poster) is much more than just a collection of its parts (in this case, characters, background colors or decorations, and objects/props). Previous work (Desolneux, Moisan and Morel 2008; Guberman 2015; Kobourov, Mchedlidze and Vonessen 2015) has implemented Gestalt principles for computer vision, graph drawing and image analysis, but our system uses them to create the poster’s composition.

To achieve a good composition, the poster should be in equilibrium. The main reference for equilibrium in composition is balance, which establishes that each figure or element inside the composition has come to a standstill and “no change seems possible” (Arnheim 1974). The two main properties that affect the balance are weight and direction.
Weight refers to a visual element’s capacity to generate tension in the composition and it is defined by taking into account some of the characteristics of the element. Some quantifiable characteristics are location (an element is considered heavier if it is further away from a poster’s center), spatial depth (an element is considered lighter if it looks closer than those that are seemingly further from the viewer), size, color (an element is considered heavier if it is brighter or more reddish), and isolation (Arnhem 1974).

However, other characteristics of a visual element like intrinsic interest (spectator preferences) and shape (how the viewer perceives object boundaries and their axes or structural skeletons) are related to human perception (Arnhem 1974), and therefore cannot be as easy to quantify. In our framework, this potential drawback is mitigated by the use of Case-Based Reasoning (CBR), which encodes in the cases these qualitative human “measurements” (opinions) and combines both the quantitative and qualitative descriptors of known posters in order to propose the composition of new posters.

The main purpose for using CBR is to allow the reuse of old solutions, in this case existing movie posters, to meet new demands, in this case new movie posters we want to generate (Kolodner 1993). In other words, using CBR allows us to retrieve and compare features of the posters stored in the knowledge base to generate a new poster following the new requirements and the Gestalt composition theory. The reason why this is useful is because in art and design it is common to use inspiration from previous and similar artifacts rather than beginning the design of a new one from scratch.

The second property related to balance is direction, which refers to the possibility of a viewer perceiving some visual elements in a composition as pointing somewhere (perhaps towards another object), creating visual lines that result from an element’s neighboring elements, subject matter, and shape (Arnhem 1974). Only the first trait is easily quantifiable, but again, CBR helps to combine these with qualitative attributes.

Balance is reachable through many strategies (Arnhem 1974) that assign weight differently in order to obtain a stable mass center. Some of them include an analysis of the symmetry axis (vertical, horizontal, or diagonal), top vs. bottom balance (where top elements are heavier or lighter than bottom elements of the same size and location) and right vs. left balance (which applies if no symmetrical balance exists because elements on the right are heavier than elements on the left, or vice versa). The calculation of mass centers (McManus, Stöver and Kim 2011), weights, distances, and correlations between objects (like similarity or overlap of color positioning) is necessary in all the mentioned strategies.

The final aspect of a good composition is a defined form for each figure. The form refers to the viewer's perception of an object regarding composition, which is not necessarily the same as the shape of the object (Arnhem 1974). Consider Figure 1, where a disc is displayed. Although the image shape is that of a disc, without this information a person could infer that the image shows two circumscribed circles, a tire, or even a donut. Thus, composition should provide enough details of each visual element to communicate its intended interpretation as clearly as possible.

Because CACARO suggests a composition for a poster, each visual element’s form can only be altered by foreshortening or overlapping. These two strategies must take into account image aspect ratio, framing, and continuity, so the element should not look distorted or amputated (Arnhem 1974). Additionally, it is convenient in posters to consider the concepts of positive and negative spaces, because the blank areas (negative space) that appear together with the objects of interest (positive space) are a “critical composition element”, as Suler (2013) points out.

### Decision Trees and Random Forests

In machine learning, classification problems can be described using decision trees. Each internal node of a decision tree contains a question (or, at its most basic, the name of a decision variable), and the set of branches departing that node represent the different potential answers to the question (or values for the variable). By answering the questions and following the branches with respect to a given instance (example) which one wants to classify, each time gaining more distance from the root node, the search becomes more and more focused, and the search space more and more reduced. Each leaf node represents a possible classification (decision, prediction, value) for the instance which is being analyzed using the decision tree.

It is very important to highlight that many different trees can be built to solve a single problem. Figure 2 shows two different kinds of “footballs”: on the left, the classic soccer ball, round with interlocking pentagonal and hexagonal shapes, usually black and white, etc., and on the right an American (gridiron, NFL) football, shaped like a pointy oval with tiny bumps on the surface, usually brown, etc.

Let us assume that a system is trying to classify a new instance that has been thrown at it as either a soccer ball (+) or an American football (-). Depending on the features we decide to analyze, the ambiguity (or lack thereof) in the values of these attributes, and even which of these parameters we use in the root node of a classification tree, we may get very different trees with the same aim, as Figure 3 illustrates.

In Figure 3 one can observe that a big advantage of decision tree classifiers is interpretability. A human being can see the decision tree and understand the reasoning behind

**Figure 1.** Illustration of a disc.  
**Figure 2.** Two types of football: on the left a soccer ball and on the right an American football.
the classifications, which is not the case with many other machine learning classifiers.

When creating a decision tree, an important question is which question or variable should be included at each level of the tree. Two of the most popular options are the use of information gain and the use of the Gini index to make the decision. Both metrics aim to build compact decision trees, since the larger a tree gets, the more it tends to under-generalize (overfit) the training data. Another approach to avoid overfitting is the use of pruning techniques which reduce the tree’s size. This might lead to lower accuracy in the tree’s description of the training data, but it leads to better generalization and therefore better results when the tree is used to classify unknown data (Shalev-Shwartz and Ben-David 2014; Kuhn 2017).

However, in CACARO, to reduce the risk of overfitting we used random forests. A random forest is a collection of different decision trees applied to the same problem. Each tree classifies an instance individually and then, through majority voting, a final classification is proposed. The trees inside a forest are all different from each other and use only a subset of parameters to make their classification. The generalization error of a forest depends on the strength of its trees and the correlation between them. Using random subsets of attributes to do this not only reduces the error but increases the classifier’s tolerance to noise (Breiman 2001).

CACARO runs the random forest algorithm five times on the same problem description with different sets of hyperparameters in each of these runs. Some such hyperparameters used in the random forest algorithm are the number of features that are analyzed, the weights of the different features that are considered and using different random seeds for each forest. The result is a list of up to five different cases that are deemed to be most relevant by the random forest algorithm (though there could be duplicates despite varying the values of the hyper-parameters). Despite the various cases that are retrieved, they are all relevant to the problem description or they would not have been the random forest algorithm’s selection. This method for case retrieval is not unique to our work. It was analyzed at length along with other alternative case-retrieval techniques in (Löw et al. 2019). In addition, this technique was found to increase the effectiveness of case retrieval in (Yang and Wu 2001). Still, even though it is not the most widely used case retrieval method, given our application domain we found it appropriate.

Once the system retrieves relevant cases through the use of the random forests, we need a way to use the knowledge implicitly held in them to propose a description that contains the main characteristics of these relevant cases represented in one general description for the new poster composition. To accomplish this case adaptation, the system uses heuristic rules based on the theory of composition.

**Heuristic Rules**

After relevant cases have been recovered, the system proposes values for attributes that were not included in the initial problem specification by abstracting a generic case from the recovered cases through the use of heuristic rules. Heuristic rules are used to find solutions for problems in artificial intelligence. These rules trade accuracy and completeness in favor of speed and performance in their decision making (Pearl 1984). These rules are often used in bundles to break up complex problems into more simple ones and to find approximations to the solution by aggregation (Rommers et al 1985).

CACARO breaks up the problem of synthesizing a resulting solution proposal from the outputs of our random forests into several heuristic rules which adjust select features from the representations of retrieved cases. This results in obtaining a single adaptation which synthesizes case properties, thus generating better poster proposals than if the knowledge held in the recovered cases was used. In other words, we consider that all matching cases might be mined for useful information in proposing the new composition, not just the top matching case.

**Implementation**

The algorithms that process a new problem and implement the case retrieval and case adaptation sub-tasks in CACARO are written in Python. This programming language provides functionality that makes it easy to process large amounts of data without writing large amounts of code, and also has many predefined libraries that simplify the processing of visual information.

Each poster in our system is represented as a separate case. Some of the information stored in each case includes values for attributes such as the number of characters from the movie shown in the poster, the type and number of objects/props that are present in the poster, the type of background included in the poster, and so on. Each case is described using the same standard set of attributes, which allows for easy comparisons among the cases and easy probing of case memory to determine the cases that are relevant given the description of a new desired poster.

![Figure 3. Two alternative decision trees for the same problem (classifying an object as a soccer ball or an American football).](image)
Figure 4 shows the general description and structure of a case stored in CACARO. Each case can be described by at most 29 attributes, including both the description of the solution and the problem requirements for a given case. The attributes used for each case are classified by Category and Type of Information.

Categories in CACARO are groups of features related to the description of a poster element. There are four categories: Description (general information about the case), Composition (elements considered for the composition of the poster), Characters (description, image and properties of each character shown in the poster), and Visualization (characteristics of the poster that affect its visual perception which are not included in any of the previous categories).

Each attribute in Figure 4 is prefixed by the initial of its category name (in the case of Characters, their attributes are prefixed by a “CH” rather than just the initial, to distinguish from Composition attributes, which are prefixed by a “C”). For example, Movie Name is an attribute belonging to the Description category, so it is prefixed by a “D”.

Consideration for Type of Information classifies the features of a case based on their role in the poster design process. The categories here are: Goal, Input Image Requirement (IIR), and Outcome. Goals are the user requirements for a poster and they contain the information that defines the main properties required in a new poster (new case). On the other hand, for the cases initially contained in CACARO’s case base (old cases), we manually assigned the values for the attributes classified as Goals based on the perceived characteristics and final appearance of the corresponding poster. This is due to the fact that the initial cases were not the solutions to problems posed by the user.

For the design of a new case (poster), values for attributes described as Goals must be provided by the user, except for the attributes marked with an asterisk, which are optional for the user to provide (for example, Feeling) if more restrictions than the minimum amount need to be specified.

An IIR attribute describes the information provided by the user which identifies and includes the images inserted for a new case. These images must be about the characters, backgrounds (scenery, locations), and/or the logo that can appear in the new poster.

Finally, attributes described as Outcome contain the descriptions of the case which are the result of the composition and design processes performed by CACARO. In other words, the role of CACARO is to fill in the values of some or all of these attributes as a result of the decision-making process that the system performs in order to solve a new poster composition problem.

Some of the attributes, like positivity, are the result of performing an analysis of the image in the poster and its goal and input requirement attributes. Therefore, the solution part of each case within the case base is fully described using only 15 stored attributes: Case Name, Movie Name, Genre, Feeling, Symmetry Type, Total Number of Characters, Color Palette, Background Image, Movie Logo, Character (Role, Class, Image, Priority, Framing) and Poster Image. Each attribute in the case base has a specific type of value. Valid values for each attribute depend on boundaries defined by the associated data type and domain knowledge.

Some of the attributes contain the data directly. For example, features like Movie Name contain string data (without allowing null values). Character Image, Movie Logo, and Background Image are PNG images. Some attributes described by integer data are Luminosity and Contrast (derived attributes), with valid values between 0 and 255.

However, most of the attributes were represented internally using bit strings, a process which allows them to be easily used in the random forest and the heuristic functions used in CACARO (which are programmed to manipulate bit strings). The encoding techniques used to represent those attributes were the one-hot encoding and binary encoding techniques, depending on the attribute. It is important to mention that binary encoding allows the description of each exemplar through one or more categories (values) at the same time, whereas one-hot encoding is intended to be used when categories are mutually exclusive (Cohen et al. 2013).

The binary encoding technique is used to represent categorical data in a binary format, where each category is associated to a position within a bit string and the bit is turned

![Figure 5. Example of the binary encoding technique in CACARO.](image-url)
on or off depending on the presence or absence of the descriptive category in a particular example. Figure 5 is an example of the binary encoding technique applied to the *Feeling* attribute (the only attribute which uses this encoding scheme in CACARO).

In Figure 5, each row shows the description of one movie poster, and both examples shown use two sentiments to describe the feeling or mood that the poster design was intended to convey (fear and anger in the case of the first movie, and surprise and sadness in the case of the second). However, a movie poster can be associated to only one sentiment or to more than two.

On the other hand, the one-hot encoding technique similarly assigns one bit position within a bit string to each possible value of a category but limits the entire bit string to only have one bit turned on (Pai, Pardawala and Potdar 2017; Harris and Harris 2013). Figure 6 shows an example of this technique applied to the Genre attribute. Again, the figure shows the descriptions of two movies, one per row. One of the movies is classified as a Romance film and the other as a Horror movie.

Each case in CACARO was stored in its own row within an Excel file stored in .csv format which facilitates loading them into the random forest code that uses scikitlearn. This Excel file represents the system’s case base. The features taken for this file are: *Poster Name (file)*, *Movie Name*, *Genre*, *Feeling*, *Symmetry*, *Number of Characters*, *Color Pallette*, *Character1 Class*, *Character1 Priority*, *Character1 Role*, *Character1 Framing* (and the same for characters 2 and 3 if they’re needed), *Background Description*, *Background Priority*, *Object Description*, and *Object Priority*. It bears mentioning that each “Priority” feature is a reference to the importance of the character in relation to the others, affecting their relative size within the poster. Also, background and object description are used to mean the presence or absence of either within a given case.

Given the fact that random forests only compare numerical values, most of the values were codified with one-hot encoding. An example of the codification proposed for one of the features, *Genre*, is as follows:

<table>
<thead>
<tr>
<th>Genre</th>
<th>Terror</th>
<th>Drama</th>
<th>Romance</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0001</td>
<td>0010</td>
<td>0100</td>
<td>1000</td>
</tr>
</tbody>
</table>

If there is no object or background then the related features in the Excel file will be assigned a value of −10, and the same occurs with characters 2 and 3 if they’re absent from a given poster.

In order to explain CACARO’s workflow, Figure 7 shows all the main blocks and step-by-step tasks that CACARO...
performs, which we describe in detail in the following paragraphs. Each input and output of each process is listed below the diagram and referenced in it with a capital letter. In the beginning, the user inputs a request to generate a new poster by giving some characteristics of the desired poster, for example the genre of the movie, a color palette, the number of the characters and some images of those characters. Once the input with the features have been introduced, CACARO translates those features into a special vector to be used as input in the random forests.

Random forests are instantiated and trained using the scikitlearn module for Python. Given the fact that each poster is a different and unique case in the case base, they can’t be grouped together to classify a new instance. Because of this fact we made the decision that the individual posters are leaf nodes in each of the decision trees.

For each design problem the system produces five random forests with different hyper-parameters (for example, different amounts of features per tree, different random seeds which allow the experiments to be reproducible and the random forests to be different from one another, and also different number of trees per forest). The leaf nodes of the resulting forests are posters which contain some of the characteristics that the user desires (included in the initial problem specification). After gathering the five results, only the set of cases corresponding to these five posters is recovered from the case base. CACARO proposes values for attributes that were not included in the initial problem specification by abstracting a generic case from the recovered cases through the use of several heuristic rules that synthesize the data held in the five retrieved cases. The following paragraph explains some of these synthesis rules.

For the number of characters, we get a range based on the average of number of characters found in the recovered cases. For the symmetry, we randomly obtain a symmetry from those in the recovered cases. For the poster’s colors, we take the top five colors present in each case (determined by measuring and comparing the total number of pixels occupied by each color) and find the mean saturation. We then do the same for each of the other four cases, and finally find the mean of the five means. An analogous process is followed in order to determine the hue. Both the new saturation and the new hue found through this process are used as a filter when deciding the color of the resulting poster(s). For the positivity, an attribute whose semantics we describe in more detail below, we obtain the median of the recovered cases and establish a range of ±12% from this value as the acceptable range of values for the proposed cases.

We think that these heuristics which combine some random decisions within limits obtained from the cases in the case base serve to give CACARO its potential for creativity. CACARO has a second set of heuristics which yield recommendations that depend on the inputs the user gave to initiate the (re)design process. These heuristics refer to the number and type of characters and their framings (the percentage of each character that appears in the poster—only face, full body, etc.—depending on the character image(s) provided by the user).

All the previous heuristic rules determine values for each of the relevant parameters in the description of a poster’s composition. These values are used in the case preparation phase to generate the resulting poster. In this phase, CACARO makes use of the information related to the heuristics and also the user information about characters. It establishes the initial position and size of the images within the poster it will propose and it also uses the character class specified by the user later, for the evaluation of the proposed composition of the poster.

To perform the poster composition, we need to consider the following premises based on the knowledge base and the visual posters considered: 1) the poster must have a background; 2) the characters do not have any specific predefined order in the composition; and 3) the size of the characters in the poster will be defined by their role and importance in the actual movie. With these premises in mind, the order of the composition is performed randomly for each proposed poster, changing the size of each character and their position within the poster. Using this technique, CACARO can find the best composition by evaluating it using the heuristic rules described above.

To assess the quality of the composition of a proposed poster, some characteristics of the image are evaluated. If one or more characteristics do not fulfill the requirements or thresholds established by the heuristics, CACARO performs the actions needed to improve the composition and meet the thresholds. These characteristics are evaluated by four rules based on Gestalt theory.

The first rule evaluates the proportions criterion, which refers to how much the character image can be resized and moved within the poster; therefore, it is not a criterion with degrees of compliance, but instead a hard requirement that the image must accomplish to be considered a good poster composition according to the evaluation of the relations between composition elements as described by Arnheim (1974).

CACARO compares the relative sizes of the different characters included in the poster and the ratio of the character’s size to that of the poster itself. It also determines if those ratios fulfill the criterion by comparing them with desired intervals. Each desired interval is different based on characters’ classes and priorities, and their values were determined empirically based on the range displayed by the relevant posters in the case base. For instance, if CACARO compares within a poster proposal a character X (class=protagonist, priority=2) with a character Y (class=secondary, priority=3), the proportions criterion will accept the size ratio of X to Y only if its value falls between 0.7 and 0.9 because those are the limits of the size ratios between protagonists and secondary characters with respective priorities of 2 and 3 within the posters that are included in the case base.

The second evaluation, related to the posters’ positivity, is based on the degree of overlap and sizing of the characters in the posters. This is done by using a pixel matrix (positivity matrix) the same size as the poster in which each pixel
occupied by a character is set to 1 and the rest is set to 0. Positivity is equal to the number of pixels occupied by a character divided by the total number of pixels in the poster. As explained in the heuristics, this rule takes into account the positivity of the recovered cases and obtains the average positivity, which is used to define the acceptable range.

The third rule stems from the concept of balance from Gestalt theory. For its evaluation, CACARO uses the selected axis of symmetry for the current composition to split the positivity matrix. This matrix shows only the position and form of the characters, which are the equivalent of the tension elements described in Gestalt theory (Arnheim 1974). But, in order to calculate balance, we need to determine the weight of each element given its position, as Gestalt theory specifies (McManus, Stöver and Kim 2011). Accordingly, we defined a weight matrix which represents the distribution of the visual tension described by Arnheim (1974), where the top-right pixel (TRP) is the heaviest part and the bottom-left pixel (BLP) is the lightest. This matrix is used for all compositions and its values are defined as 0.1 for BLP and 1.0 for TRP.

The weight matrix is multiplied by the split positivity matrix to quantify the tension generated by the two halves of the poster. Then, the tension difference (in percentage) is calculated. If its value is less than 10% the poster is considered as balanced.

Lastly, the system performs face detection of the characters in the poster to evaluate whether the composition of the characters is correct or a resize or translation is needed. If the faces of two characters overlap by at least 20% of the rear face, one character is moved to eliminate the excessive overlap. CACARO accomplishes this by implementing YOLO’s neural network architecture (Redmon and Farhadi 2018), trained with 600 hand-labeled images with faces in them, taken from the WIDER FACE dataset (Yang, Luo, Loy, and Tang 2016).

Regarding the rules that CACARO uses to evaluate the posters, each one assigns a degree of quality according to a specific criterion. These measurements are averaged to obtain the overall quality of the poster, which can be assigned from 0 to 100%. In this way we can establish a minimum quality threshold needed to accept a composition for the poster as a valid solution.

Regarding the proportions criterion, the degree of quality is measured according to the percentage of size ratios that fall within the acceptable interval of values. We set CACARO’s threshold for this criterion at 100%, which means that all the size ratios must be within the corresponding interval in order to achieve an acceptable quality solution.

The positivity rule guarantees the poster’s positivity is within the acceptable range set by the rule. We consider the difference between the average positivity and the proposed poster’s positivity to define the degree of quality of the poster according to this criterion by applying the following formula:

\[
\text{Positivity quality} = \left(1 - \frac{\text{poster’s positivity} - \text{positivity average}}{\text{positivity average}}\right) \times 100%
\]

We set CACARO’s threshold at a minimum of 88% to ensure quality but also to allow variability and creativity in the generated posters.

The measurement of the degree of quality for the last two criteria (balance and face detection) is similar to the previous one, but instead of the difference in positivities, balance uses the difference in tensions and face detection uses the percentage of overlap between faces. We set their threshold levels in CACARO at 90% and 80%, respectively.

As explained, the four criteria need to be met so a proposed solution can be considered “acceptable”. In order to fully converge CACARO first evaluates each criterion in the order presented in this paper. If a criterion was not met, CACARO makes the previously explained changes and again evaluates each criterion. If any of the previously met criteria is not met anymore due to the modifications, CACARO will start making further modifications, and reevaluating, until all criteria are met.

As a result of the previous implementation phases, CACARO provides one resulting poster which must fulfill the user requirements and must be generated by evaluating the metrics and meeting the criteria described above. CACARO performs the poster composition process based only on Gestalt theory (which might not be in accordance with criteria from other design theories).

However, CACARO also generates several additional alternative compositions for the poster. This allows the user to perform a visual comparison to complement the algorithmic decisions made by the system previously. The user can then decide which is the best alternative. This framework can also improve creativity through the variety of options present among the proposals. A saturation and value filter is applied to the final proposals in order to get similar saturations and values to the retrieved cases.

Experiments and Results

We tested CACARO with several problems to evaluate the quality of the results. In this section, however, we only present three of these experiments, one of them fully detailed and the other two just presented through their inputs and outputs.

In the first test problem, the goal was the redesign of the poster for “The Prestige” (Touchstone Pictures 2006). The most important input requirements were “Drama” (as Genre) and the images and features (Class and Role) of the three main characters of the movie. Based on all the inputs, the random forests suggested that five most relevant cases are “Men Of Honor” (Twentieth Century Fox 2000), “The Shawshank Redemption” (Columbia Pictures 1994), and “Kick-Ass 2” (Universal Pictures 2013), the last one retrieved three times. These retrieved posters can be consulted
Afterwards, the heuristics portion of the system established the acceptable ranges for the proposals’ qualitative features such as positivity (from 58.95% to 75.03%) and type of symmetry (as diagonal).

Given the previously mentioned features and inputs, CACARO initialized the characters’ positions and sizes in one of the proposals as shown in Figure 8. CACARO then began evaluating the proposal, starting with the relative proportions amongst the characters and the poster until they were within the acceptable limits. Afterwards, the system started making adjustments to reach the required positivity level, reaching 60.13%. Then, given the type of symmetry, CACARO tried to compensate the weights of each side of the symmetry axis until it achieved a balance of around 47%.

Finally, CACARO checked for overlapping faces and corrected positions when needed and applied the corresponding filter. It is important to notice that the evaluation process is sequential and iterative: when achieving an acceptable level of positivity, for example, the proportions criteria could end up being out of range even if it hadn’t been originally. This forced CACARO to reevaluate and adjust the proposal until an acceptable value for all the criteria was reached, giving us the final state of this proposal as shown in Figure 9.

The adjustments consisted of resizing the characters (for the proportions and positivity criteria) or repositioning them (for the positivity, balance, and face overlap criteria). The magnitude and/or direction of these adjustments were determined based on optimizing the poster’s composition together with a random component, which allowed CACARO to avoid stagnation and to propose a variety of possible and creative solutions.

The degree of quality of this poster, obtained by the metrics previously detailed, were 100% for the proportions criterion, 89.76% for positivity, 97.00% for balance and 98.23% for face detection, resulting in an overall degree of quality of 96.24%.

The second test problem was the redesign of the poster for “Raiders of the Lost Ark” (Paramount Pictures 1981). The most important input requirements were “Action” (as Genre) and the image and features (Class and Role) of the main character. Therefore, CACARO made a poster composition based on one single character whose size and position determined the poster’s quality. The proportions, positivity, balance, and face detection criterion obtained quality values of 100%, 92.83%, 90.26% and 100%, respectively. The final poster, shown in Figure 10, obtained a quality level of 95.77%.

The final problem had as goal the redesign of the poster for the movie “When Harry Met Sally” (Columbia Pictures 1989), where the input requirements included the two main characters and the “Romance” genre. The proportions criterion and the face detection had a 100% value of quality,
while positivity and balance obtained 93.45% and 92.10%. The quality level for the final poster, shown in Figure 11, was 96.38%.

It is important to keep in mind that the whole process from generating an initial composition to getting the final result involves the encoding techniques previously described. Each poster proposal is described by its features (number of characters, genre, feeling, etc.), and each feature is represented by numbers and bit strings where each bit string follows the restrictions imposed by the one-hot or binary encoding representation. Those feature representations form the basis for generating different visual proposals.

Discussion, Conclusions and Future Work

In this paper we have described CACARO, a system that proposes movie poster compositions. The algorithms that CACARO follows in order to perform its task are centered around case-based reasoning, augmented with random forests (for case retrieval) and heuristic rules (for case adaptation). The combination of some of the random choices present in the system's process model and the limits imposed on them by its domain knowledge as embodied in the case memory provide the inspiration that the system uses in order to propose new poster compositions and are the source of its potential for creativity.

The current version of the system has some limitations which we plan to address in the future. Some of these are, for example, the ambiguity of some features. For instance, the feeling (mood) that the poster transmits and the role of a character in a movie may be subject to interpretation. The current implemented version of the system only has in its case base posters for movies in the horror, drama, romance, and action genres, which limits the range of values of the mood variable within the case memory, and this may affect the results.

On the one hand, a wider variety of genres may provide the system with the capability to produce poster compositions that are more "fine-tuned" to the particular specifications of a new problem. On the other hand, the more genres that are included, the greater the possibility for overlap/ambiguity between genres, and thus the greater the chances that the system's proposals might not be as good as possible. We plan to perform ablation experiments with a more complete case base and reduced subsets of the same case base (produced by extracting some genres) to see the effect this has on the types of solutions proposed by the system. There is also the possibility of talking with a panel of experts (composed of designers and filmmakers) to evaluate CACARO’s results and try to tweak the poster generation process based on the experts’ feedback. Another limitation is that our cases are structured in such a way that only up to three characters per poster may be included in the case description, and if there are less the values assigned to the absentee may disproportionately affect the random forest classifiers because we assign them a −10 value.

The size and scope of the case base (which contains 103 cases) might affect the system’s predictions and accuracy. This results in some edge situations in which all the forests may return the same solution, and this doesn’t leave room for changes to be made during the case adaptation phase of the system.

We intend to implement an evolutionary algorithm in order to make CACARO even more creative. The algorithm will start with a population consisting of the five cases resulting from the random forests plus some randomized individuals. Then, through many generations of mutation and crossover operations on the individuals we will generate more varied proposed compositions. To ensure the quality of each new generation our evaluation function will discard bad individuals and eventually will help converge on a result which will be the proposed poster. This will require that each individual in each generation be evaluated by its “phenotype”, i.e., the poster itself. Thus, the system will require more computational power for image processing than the current version. We will also have to reformulate certain aspects of the evaluation function. Previous work (Gómez de Silva Garza and Maher 1999) has shown that evolutionary algorithms can provide good results when used for the adaptation of cases.

Acknowledgements

This work has been supported by Asociación Mexicana de Cultura, A.C.

References


Harris, D., and Harris, S. 2013. Digital design and computer architecture. California: Morgan Kaufmann.


MIST: “You Play, I’ll Draw”

Juliana Shihadeh and Margareta Ackerman
Department of Computer Science and Engineering
Santa Clara University, Santa Clara, California
{jshihadeh,mackerman}@scu.edu

Abstract
What if creative machines could show the story told by a live orchestra? In this paper, we propose a vision for a co-creative orchestra performance partner that enlivens this classical art form and has the potential to increase audience engagement and reach broader audiences. We present MIST, an early version of our vision, which creates real-time visualizations as live music is performed, and showcase it in a violin performance.

Introduction
Collaboration between humans and machine opens the possibility of creative entertainment that neither humans nor machines could create alone. We propose that human-machine creative collaboration can be used to enliven classical art performances. Orchestral concert attendance is on the decline, from 11.2% in 2002 to 9.1% in 2008 (Williams and Keen 2008). Through a perhaps unlikely juxtaposition of the old and the new, Computational Creativity may offer an opportunity to enliven engagement in classical performance.

Music is multidimensional, carrying structural features such as tempo, tone, pitch, range, and rhythm (Schaefer 2017). This provides ample information to extract and decrypt, giving the opportunity to translate the musical data into illustrations that unpack what music is telling through sound and bring forth a new dimension to the performance, allowing for deeper audience engagement.

In this paper, we present a vision for a novel form of collaboration between humans and machines, where the computer agent creates a visual narrative representation of music performed in real time - different every time the music is played. The machine partner would interpret the music, creating a visual story inspired by it, with the aim of deepening audience engagement though this visual medium.

One of the primary challenges with translating live sound into illustrations is real-time analysis of the music. There are many ways to visualize illustrations that represent the story told by an orchestra performance. Here we initiate this line of research, presenting an initial version of MIST where music is translated into sequences of symbols and emojis

1It is worth noting that due to their central role in youth culture, emojis have been incorporated into and inspiring art for to showcase real-time music extraction and illustration with smooth transitions between each music-to-illustration captured. We demonstrate the current version of the system through a collaboration with a human violin player.

In addition to presenting an early version of MIST, we discuss our vision for this line of work, proposing a partnership between an orchestra and a system illustrating the story conveyed through the live music (See Figure 1 for an illustration for the vision for this work).

Background
The idea of utilizing creative systems to enhance live performance of human artists has been applied in other manifestations in music and dance. For example, in live dance performance, ViFlow was used to create visuals in response to a dancer’s movement, showcased behind or below the dancer during a performance (Brockhoeft et al. 2016). Other work related to integration of music and visuals includes (Chen et al. 2008), who created a system for the integration of user-provided music and photographs through an emotion-based approach.

Improvisation systems have also been applied to live performances. For example, the musical composition system Iamus (Ball 2012) built off the Melomics hardware (Diaz-Jerez 2011) has been used to generate music in real-time, played by an orchestra or human musicians. Robert Keller’s improvisor has been used for musical human-machine improvisation trading (Putman and Keller 2015). Likewise, Andrew Brown’s (Brown 2018) musical bot partnered with a human pianist to play a live improvisational duet and Weinberg et al’s (Weinberg and Driscoll 2006) system Haile analyzes music in real-time and plays percussion instruments alongside the musicians.

MIST offers a new form of creative interaction between humans and machines, giving live musicians the opportunity to enrich their performances with improvisational visuals to complement the music, while giving audiences new dimensions through which to experience the performance.
Figure 1: An illustration of the human-machine collaboration proposed in this work: While an orchestra plays, an animated story based on the live music is automatically created and rendered onto a screen. This offers a visual dimension to the orchestra experience and stands to increase audience engagement.

MIST

Our initial version of MIST is an interactive system that analyzes music in real-time and creates symbol-based moving illustrations using information extracted from the music. We create two different visualization styles, one which consists of a sequence of emojis created on the fly in response to the music, and another which generates a sequence of colored stars. A demonstration of MIST interacting with a live violin player can be seen here: https://youtu.be/sFeHJ6DBXcQ.

Three Elements of a Live Music Illustration Systems

We propose three elements that a creative system that illustrates live music should include. These elements can enable a system to capture and represent the essence portrayed in the music.

1. Extract musical elements such as dynamics, pitch, beat, etc.
2. Listen and react to music played in real time
3. Convert the music to visual illustrations

The Emoji and Colored Stars prototypes demonstrate how features in music can be extracted to draw illustrations in real-time.

Emojis version:

As music is played, Emojis print out based on the audio’s amplitude and pitch. We use Librosa’s melspectrogram to determine amplitude and CREPE to extract frequency to determine octave. To be able to extract the amplitude and pitch of the music in real-time, we use SoundDevice to read audio in real-time, save to a wave file and then open it as if it was already pre-recorded. We divide amplitudes and octaves into ranges to map to an Emoji, first categorizing by amplitude and then the octave.

Softer amplitudes are mapped to emojis like flowers, leaves and hearts. Mid-range amplitudes focus on emojis that represent the “peak” of a story such as revolving or broken hearts. Higher-ranges of amplitude focus on a reflective, ponderous feeling as the aftereffect to the “peak” with clouds and umbrellas used to represent a rainy day where someone might be lost in thought or a waxing crescent moon to represent a new start, since high amplitude could also indicate an enthusiastic reflection of someone looking forward to the future.

Within amplitudes, we pick the final emoji based on the octave. We assign lower octaves emojis with a feeling of sadness or questioning of hope such as a flying balloon symbolising “leaving” or a seed representing “questioning hope” as one cannot really know how a seed’s growth will turn out. Higher octaves range from showing emojis giving a feeling of deep sadness or excitement depending on the amplitude using emojis like clouds or bees.

Understanding the emotion music portrays requires insight into the complex relationship between musical elements. Naturally, illustrations can vary vastly in the way they tell the story of a piece of music and one mapping offers one such option. Figure 2 is of a segment of Vivaldi’s Winter 1st Movement where the solo violinist is playing in a high octave with instruments playing softly in the background (Lockey 2017).

Please note that rather than creating static images, MIST generates the sequences of emojis in syncrony with the music, and as such is best viewed in video form. We

---

3https://pypi.org/project/crepe/
4https://pages.mtu.edu/suits/notefreqs.html
5https://python-sounddevice.readthedocs.io/en/0.4.1/
6https://www.youtube.com/watch?v=JkP7sIc9aM
present a demo of it based on a live violin performance of excerpt from Schindler’s List Theme at https://www.youtube.com/watch?v=A5p1R9gsQQU. Note how often broken hearts come up in to capture the sadness that’s heard in the music and how in one of the higher octave scenes there are clouds and small umbrellas. We also provide a demo based on a recording of Vivaldi’s 1st Winter Movement: https://youtu.be/-q9VRZ8nmcg. Note how fall leaves come up a lot in steady scenes where the octave is low and not very loud, while broken hearts come up when the scene intensifies even if it remains in a lower octave.

Colored Stars:
Tjoa (Tjoa) created a real-time spectrogram with black asterisks printing on a white background as an example to build on for further real-time music analysis. We built onto it so that as music is played various colored asterisks print out depending on the audio’s amplitude with a black background used to enhance the final display. We use Librosa2 and PyAudio8. We only worked with a spectrogram for this version following Tjoa’s work9 so we did not need to save to a wave file to extract more features, but this limited what features we could work with to map to colors. Using ANSI text color coding, from piano to forte amplitude is divided into 4 ranges mapping to blue, magenta, yellow, and red respectively. Red represents intensity, forte, and blue represent mellowness, piano. Thus the softer the volume the warmer the colors and the less stars there are, while the louder the volume the bolder the colors and the more stars there are to represent all the layers of sound waves the system is picking up on. Figure 3 shows an image of the colored stars in response to Vivaldi’s Winter 1st Movement of a segment captured of multiple instruments playing at the same time at a fast tempo10 (Lockey 2017).

As with the Emoji version, this version of MIST also prints the sequence of symbols synchronously with the music, and is best viewed in video form. We present a demo of it based on a live violin performance of excerpt from Schindler’s List Theme at https://www.youtube.com/watch?v=sFeHj6DBxQ. Note how the number of stars change in sync with the many changes in volume. Especially with much softer notes, the stars decrease even more such as on held-out notes where the sound fades toward the end. While, in more intense spots stars fill the screen with various colors to capture the different amplitude ranges. We also provide a demo of it based on a recording of Vivaldi’s 1st Winter Movement11 (Lockey 2017): https://youtu.be/ULAnbN2BeI. Note how in parts where multiple instruments are playing there are more stars displayed on the screen, while when the solo violinist is playing a staccato scene at 2:28 the stars are sparse and scattered.

Impact
In this section, we discuss the impact of MIST and its potential for partnering with live orchestras. We discuss how MIST can improve audience engagement and make orchestra performance accessible to wider audiences.

Strengthening Engagement
Art and emotions are closely linked (Silvia 2005). A study conducted on music-evoked emotions by examining functional magnetic resonance images proved that music evokes emotion, producing a variety of emotions depending on the music’s style (Schaefer 2017). Stronger emotional connection was shown to increase attention (Tyng et al. 2017). Extracting music’s attributes allows MIST to capture the music’s emotions through illustrations, providing audiences audio and visual sources to connect with, which can strengthen their emotional connection.

Furthermore, interest initiates motivation which in turn increases engagement (Murayama 2018)(Tyng et al. 2017).
Figure 4: Images illustrating our vision for the vintage drawing style in which a mature version of MIST would be able to create scenes based on live music. Public domain images.

If the introduction of corresponding visualizations interests people, it has the potential to contribute to retaining or possibly even growing attendance at orchestra concerts. Some who may not have been interested in orchestra performances may be interested in the audio-visual experience.

There are many ways the story within a music can be visually represented. Our vision for what a mature version MIST would be able to draw are scenes illustrated using a vintage drawing style shown in Figure 4.

A New Kind Of Performance

Future versions of MIST can bring out the more complex visuals conveyed through the music. Take, for example, Vivaldi’s 1st Winter Movement\(^\text{12}\) (Lockey 2017) It opens with a lower octave evoking a feeling of suspense as it builds up the suspense reaching a higher octave that makes you feel as if you are in the midst of a storm with winds picking up. Shifts in music trigger emotion, which in turn can help you mentally visualize the music (Taruffi and Küssner 2018), (Schaefer 2017). Instead of just seeing a mental image, MIST could help showcase the story visually as a part of the performance. It has been proven seeing art has a positive impact on one’s well-being. Neuroimaging studies demonstrate an immediate emotional response to artwork associated with circuitry involved in pleasure and reward (Mastandrea, Fagioli, and Biasi 2019).

Greater Access for Individuals with Hearing Disabilities

Over five percent of people, 466 million worldwide, have disabling hearing loss. (Organization 2021). By turning an orchestra’s performance into illustrations that capture the music, hearing impaired individuals could understand what story an orchestra performance is sharing with its audience.

A collaboration of MIST with a live performing orchestra can be a way for anyone who struggles with deafness to be able to connect with an orchestra performance. Even more so, a smaller home version of MIST could allow hearing-impaired individuals to connect with any music.

Conclusions

We offer a vision for a co-creative system that can bring to life the story conveyed through live music, proposing a new performance paradigm where live orchestras are accompanied by MIST drawing illustrations in reaction to the music. Our initial version of MIST extracts musical elements which are subsequently turned into an illustration, all in real time. Scaling the system into the full version is left for future work. Future version will consider other forms of visualization. However, due to the imaginative and divergent nature of visually interpreting music, there is a wide range of meaningful visuals for any musical piece. As such, any system would inherently represent limited visual meanings, allowing for a wide variety of approaches to this creative transformation. Furthermore, there are a multitude of approaches for extracting emotion from music (for example, in addition to the approach presented here, another method is explored in (Hevner 1936) and (Chen et al. 2008)).

In partnering MIST with live orchestras, we stand to increase audience engagement by opening two sources of information for audiences to emotionally connect with. Furthermore, we open the possibility of engaging with new audiences. This initial version considers western classical music, however, the vision presented here can be applied to a wide range of musical styles. A MIST-orchestra opens new horizons of creative engagement enabled through human-machine creative collaboration.

References

Ball, P. 2012. Iamus, classical music’s computer composer, live from malaga.


Tjoa, S. Real-time spectrogram.


GANlapse Generative Photography

Simon Colton$^{1,2}$ and Blanca Pérez Ferrer$^3$

$^1$ SensiLab, Faculty of IT, Monash University, Melbourne, Australia
$^2$ Game AI Group, EECS, Queen Mary University of London, UK
$^3$ Etopia Center for Arts and Technology, Zaragoza, Spain

Abstract

We describe the incorporation of text-to-image generative deep learning techniques into an art practice for making video pieces akin to time-lapse photography. We show that the process can be suitably controlled to find a latent vector able to generate an appropriate image, construct nearby vectors for similar images and interpolate between them to produce video pieces. We describe the process, how this fits into the GAN-art movement, and the cultural impact of this work in terms of an online and physical art exhibition in the Etopia arts and technology centre in Spain.

Introduction and Background

A generative adversarial network (GAN) is a pre-trained neural model able to generate content like images, audio or text. Training a GAN is done in conjunction with a critic model which rates how realistic the generated content is when compared to a dataset of target content (Goodfellow et al. 2014). The two networks start off being poor at their tasks, but force each other to improve until the generator can produce fake content that can (often) be passed off as real. GANs take a vector of floats known as a latent vector as input, propagate this forward through the network and produce numerical outputs interpretable as content, e.g., an image. Conditional GANs also take a class vector which enables them to produce different types of content. For instance, the GAN employed in the research described here is called BigGAN (Brock, Donahue, and Simonyan 2019), and normally takes a 1-hot vector which dictates the ImageNet category to generate an image for.

There has been an explosion of GAN-generated visual art in the last five years, as digital artists have adopted the technique due to the quality, variety and surprising nature of the output from GANs. As an indication of the impact GAN art is having, major auction house Sotheby’s held a recent sale which included GAN-generated pieces from Anna Ridler and Mario Klingemann. GAN artists produce images and videos by using pre-trained models or training their own, then searching for latent vectors, often randomly or using non-standard techniques described as active divergence in (Berns and Colton 2020). To the best of our knowledge, GAN-generated art has yet to be formally studied or categorised. The variety and quality of imagery has increased greatly over the years, but there are some commonalities in the majority of GAN-generated images, namely an abstract, dream-like quality in which the content doesn’t bear scrutiny, i.e., is not particularly coherent on close inspection. Alternatively, GANs can produce photo-realistic images like faces (Karras et al. 2021), and these have also been used in artistic settings.

A new approach to finding latent vectors for GANs has been developed by members of an online tech/art community recently, sharing their code through Colab notebooks (Bisong 2019). The approach employs the CLIP pair of pretrained models (Radford et al. 2021), which both calculate a latent vector encoding of a given input. For the first CLIP model, the input is a piece of text, and for the second, it is an image. CLIP has been trained so that the latent encoding of pairs (image, text) where the text appropriately describes the image – or vice versa, where the image is an appropriate response to the text – have a lower cosine distance than pairs where the match is not appropriate. CLIP was trained on 400 million (text, image) pairs scraped from the internet, and has captured a broad and deep understanding of the correlation between text and images.

The task of text-to-image generation (Agnese et. al 2019) involves producing images to somehow reflect a given text prompt $T$. One approach involves calculating a score for the appropriateness of a generated image, $I$, in terms of the cosine distance between the CLIP encodings for $T$ and $I$. Hence, a search can be undertaken to find a latent vector input to a GAN which produces an image $I$ with a CLIP encoding as close as possible to the encoding of $T$. Note that it is also possible to search for a generated image to match a given image or some combination of text and image. Various search techniques have been tried in the community, with the most successful being backpropagation, i.e., starting with a random latent vector and training it by propagating the value of a loss function based on CLIP. Such an approach was implemented into a Colab notebook called The Big Sleep by Ryan Murdock$^2$, which uses CLIP to search for latent vector inputs to BigGAN.

We have found the Big Sleep technique remarkably good for a range of text prompts, ranging from specific content to vague moods, as described in (Colton et al. 2019), where de-
tails of a modified version of the approach we implemented are given. For our purposes here, we need to know that the process can be parameterised by (a) the text prompt (b) the starting point which can be either a random or a given latent vector and (c) the learning rate for optimising the latent vector. We describe below how we have used the text-to-image technique to produce examples of a novel type of generative art form we call GANlapse videos. We describe the process in terms of the usage of text-to-image generation, GAN interpolation and video post-production. Following this, we describe an art exhibition of generated still images and GANlapse videos at the Etopia Center for Arts and Technology. We conclude with some thoughts on GAN art, how CLIP can bring further autonomy to GAN generation, and future research and art projects we have planned.

GANlapse Video Generation

We define a GANlapse video as akin to a time-lapse video but with the still images comprising it produced by a generative adversarial network rather than taken with a camera in a real-life scenario. By akin to a time-lapse, we mean that (a) the subject material is similar to that normally in time-lapse videos, i.e., looking as if it comes from reality, rather than an abstract or more artistic rendering and (b) the perceived time period covered is similar, for instance a construction site over the duration of a year, a skyline over the period of a day or the seasonal life and death of a plant, etc.

The production of GANlapse videos is in three parts: firstly a series of images is generated to reflect a given text prompt and then specialised versions of some of these images are further generated as keyframes for a video; secondly, multiple series of images animating the traversal from one image to another are generated, using interpolation over the keyframe latent vectors; and thirdly, a set of traversal image sequences is collated and post-processed into final video pieces. In this manner, we have produced 30 different architecture GANlapse videos and 3 featuring flora and fauna.

Text to Image Generation

To start each GANlapse project, we chose a particular subject material which might traditionally have been the subject of a standard time-lapse photography piece, and then derived a text prompt with which to generate images. For instance, in one project, we employed the content text prompt "A beautiful modernist building in the countryside" and used the modified CLIP-guided BigGAN process mentioned above to generate 200 images, using four GPUs for around 2 hours, with a learning rate of 0.08 and a random latent vector as starting point. Some experimentation with the text prompt was required to produce suitable images for each project, e.g., replacing "beautiful" with "serene".

We cherry-picked some suitably high-quality images and for each, we retrieved the BigGAN latent vector pair, \( l \), responsible for its output. For each image, we then produced a series of keyframe images by running the text-to-image process again, but starting with \( l \) rather than randomly, using a learning rate of 0.01, so the image produced doesn’t stray too far from the original. Continuing the modernist build-

Figure 1: Keyframe images for (a) a modernist building project and (b) a flower life-cycle project.

Table: Modifying the text prompt used as an example, we used the following modifier prompts to produce keyframes for a four-seasons GANlapse project:

<table>
<thead>
<tr>
<th>Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td>(i) “Covered in snow”</td>
</tr>
<tr>
<td>(ii) “Spring flowers”</td>
</tr>
<tr>
<td>(iii) “Summer sunshine”</td>
</tr>
<tr>
<td>(iv) “Autumn leaves”</td>
</tr>
</tbody>
</table>

Again, some experimentation with text prompts was required to find suitable keyframes for each project, and often we employed multiple small-yield sessions with slightly different prompts in parallel, to harvest good results.

As described below, the keyframe images become the basis for the interpolation videos which are ultimately the raw art material for the final productions. In other projects, the keyframe images were not produced by fine-tuning an image starting with its latent vector and using modifying prompts. Instead, multiple different content prompts were used, e.g., in a project to produce a GANlapse video of the life-cycle of flowers, we used seven content prompts as follows:

<table>
<thead>
<tr>
<th>Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td>(i) “Foliage”</td>
</tr>
<tr>
<td>(ii) “Beautiful small yellow flowers”</td>
</tr>
<tr>
<td>(iii) “Large yellow flowers”</td>
</tr>
<tr>
<td>(iv) “Dying yellow flowers”</td>
</tr>
<tr>
<td>(v) “Dead yellow flowers”</td>
</tr>
<tr>
<td>(vi) “Dead foliage”</td>
</tr>
</tbody>
</table>

Once the keyframe images were generated and chosen, in order to produce suitable videos, we ordered them as a time-lapse storyboard, e.g., a building’s surroundings changing over four seasons, or a flower dying and being re-born. Example keyframe image orderings for both a modernist building and flower life-cycle project are given in figure 1.

GAN Interpolations

GAN interpolation is a common technique used for inbetweening two keyframes \( k_1 \) and \( k_2 \) to produce suitable intermediate frames. Here, a sequence of images is produced by generating an GAN image with latent vector pair \( ((1-t) \times l_1) + (t \times l_2) \) as \( t \) varies via some increment from 0 to 1, where \( l_1 \) and \( l_2 \) are the latent vector pairs responsible for producing \( k_1 \) and \( k_2 \). Such interpolation between keyframe images produces pleasing and tranquil animations similar to traditional time-lapse pieces, when the generated images are compiled into a video. We experimented with in-out easing functions to slow down the movement towards and away from the keyframes to make the video subtly smoother. Such easing involves applying a function to \( t \) before employing it to calculate the latent vector pair for the GAN, e.g., substituting \( t \) with \( 16t^5 \) for quintic easing.
We adopted the practice of making the last in the sequence the same as the first, so videos can loop, and we experimented with randomly perturbing the latent vector by small amounts (up to 0.04 standard deviations away from the original) before generating the images. For the architecture videos, such perturbations produced a jittery effect commonly seen in traditional time-lapse photography, due to slight camera movements. In the case of the flowers, this effect also gave an impression of them being delicately blown in the wind. We also found that pausing on a keyframe for a number of frames worked well with perturbation for the flowers projects. While maintaining 30 frames per second for smooth videos, we experimented with the number of frames between keyframes, in order for change to be visible at all times, yet the tranquility of the pieces to be upheld.

When inbetweening the second type of keyframe sequences, namely where a set of image generated from different content prompts are used, we had two difficulties. Firstly, the distance between the latent vectors was often quite large and so the visual change from one intermediate frame to another became disorientating. Secondly, the interpolation points occasionally went near to a one-hot class vector, which generated an image in the original ImageNet categories that BigGAN was trained on, usually depicting a dog or a bird, which ruined the animation. To solve both problems, we simply calculated in advance the cosine distance between the BigGAN latent vectors of two potential keyframes and avoided using any pairs where this distance was significantly larger than the average. For each project, we varied the usage of easing, perturbation and pausing and were able to produce (subjectively) aesthetically pleasing time-lapse style animations of between 30 and 60 seconds, which acted as the art materials for the final stage.

**Video Post-Processing**

Due to the nature of the processing, GAN-generated images tend to be square, which is rarely the case for traditional time-lapse pieces. Moreover, as described below, we were commissioned to produce GANlapse videos for an exhibition involving screens with a 16:9 rectangular aspect ratio. For these reasons, and also to be a little more distinctive in the GAN-art world, we post-processed the videos to both change their shapes and make them more sophisticated. In particular, with the architecture images, we noticed that often the buildings depicted were cut by the edge of the image. This gave us the opportunity to mirror the images down the horizontal and produce a widescreen video, as depicted in figure 2(a) for the original architecture image of figure 1(a). For others, rather than mirroring the entire image, we mirrored one or both sides slightly in order to produce a wider, rectangular aspect. We also cropped where appropriate.

For two other projects, we collaged multiple interpolation videos into a more sophisticated piece. For instance, for a piece depicting the life-cycle of wildflowers called *Les Fleurs de Vie*, we combined five interpolation videos into a single one, as shown in figure 2(c). Each of the sub-windows of the piece provides a portal onto a different coloured flower at a different stage during its life-cycle. To complete the piece, we added a glassy border and table-top

![Figure 2: Still images from: the Arquitecturas Imaginadas GANlapse pieces (a) *The Museum* and (b) *The Glasshouse*. Still images from the (c) *Fleurs de Vie* and (d) *Mighty Oak* GANlapse video pieces, published on the Hic et Nunc NFT platform (hicetnunc.xyz/simoncolton).](image-url)
reflection effect. For another piece, entitled *Mighty Oak*, we again combined four different interpolation videos depicting different views of foliage during different seasons, again adding a border and reflection. For the post-production, we used our own image manipulation software and the ffmpeg software package ([ffmpeg.org](http://ffmpeg.org)) which, among many other things, can combine still images into MP4 videos. Still images from some of the GANlapse videos are given in figure 2, highlighting the mirroring, collaging, bordering and reflection techniques employed.

**Arquitecturas Imaginadas**

The New European Bauhaus[^3] is a cross-cultural initiative providing a platform for innovative thinking around living spaces, sustainable living and quality of life. One of the participating organisations is the Etopia Center for Art and Technology in Zaragoza, Spain. We were commissioned by Etopia to create an exhibition entitled “Arquitecturas Imaginadas” (Imagined Architectures) as part of the New European Bauhaus initiative. The first author of this paper provided the images/videos for the exhibition, for which the second author (working at Etopia) is the curator and mediator. The online version[^4] of the exhibition runs from 1st June 2021 to October 2021 and comprises 13 GANlapse videos and five sets of 6 still images, as before generated via quotes from well-known female architects: Julia Morgan, Christine Lam, Pascale Sablan, Zaha Hadid and Marian Kamara. Screenshots of the online exhibition are given in figure 3. The physical exhibition started on June 30th 2021 and runs until October 2021. It comprises the following:

- 16 printed images generated using the CLIP-guided BigGAN process above, using as text prompts quotations about architecture from 4 female architects.
- 21 different GANlapse videos split over four 24in screens, two 43in screens and a 2m by 1.5m video wall.[^5]
- 27 different GANlapse videos an a two-wall 8m by 5m exterior media facade.

Photographs from the physical exhibition are given in figure 4, along with photographs from the media facade. 30 full resolution GANlapse videos, 54 still images and a blogpost further describing the making of the videos from the exhibition are provided here: imaginative.ai/wp/imagined-architectures

**Conclusions and Future Work**

GAN artists have successfully cultivated their moderately abstract, dream-like aesthetic and promoted the process of serendipitous, often random usage of generative processes (Berns and Colton 2020). They regularly produce beautiful artworks which impact the art world. The work and resulting exhibition described here has been an attempt to broaden the usage of GANs in art beyond the current aesthetic. In particular, we wanted to see if it was possible to...
Figure 4: Physical installation of Arquitecturas Imaginadas, inside the Etopia centre and on the media facade outside it.

Acknowledgements

We would like to thank the anonymous reviewers for their insightful comments. We would also like to thank the team at Etopia for their hard work and expertise in staging the Arquitecturas Imaginadas exhibition.

References


Adversarial Learning of Expectation and Surprise: Experiments with Geometric Shapes

Oseremen O. Uduehi
School of Electrical Engineering and Computer Science
Ohio University
Athens, OH 45701
ou380517@ohio.edu

Razvan C. Bunescu
Department of Computer Science
University of North Carolina at Charlotte
Charlotte, NC 28223
razvan.bunescu@uncc.edu

Abstract

The ability to produce surprising outputs is a cornerstone of creative behavior. In this paper we propose deep learning architectures that are trained in an adversarial setting to learn patterns of expectations and surprise from data. We introduce benchmark datasets of geometrical shapes that represent well defined patterns of violations of expectations, and use them to verify empirically that the separation between learning of expectations and learning of surprise is essential for achieving good generalization performance.

Introduction and Motivation

Deep generative models based on Generative Adversarial Networks (GAN) (Goodfellow et al. 2014), Variational Auto-Encoders (VAE) (Kingma and Welling 2014; Rezende, Mohamed, and Wierstra 2014), and Transformers (Vaswani et al. 2017) have achieved impressive success in domains ranging from image generation, to music composition, to text generation. The ability of these models to generate highly realistic samples in very diverse modalities is made possible by their capacity to efficiently learn high dimensional distributions from large amounts of training samples. Once trained, the models generate new samples through a random sampling procedure, such as sampling a vector of values from a Gaussian to use as input to the GAN’s generator, or sampling the next word to generate according to the distribution computed by a Transformer-based language model. When trained on raw representations of real world objects or artifacts, this one-model sampling procedure preserves the structural constraints of the training domain, generating very realistic samples in terms of how they look (e.g. images of faces), read (e.g. prompt-based text generation), or sound (e.g. symbolic music). The generated outputs are also new, in the sense that they do not match any object from the training data, and the model can be used to generate a virtually infinite number of new outputs simply through its random sampling procedure. While generated samples are new, they cannot be said to be original or surprising. Being sampled from the learned distribution, they still resemble objects from the training data to a very large extent. As such, while impressive at generating objects that closely resemble, but do not reproduce exactly, the training samples, current approaches to training generative models are severely limited in their ability to generate novel or surprising outputs. Given the importance of novelty and surprise in the design (Yannakakis and Liapis 2016) and evaluation (Grace and Maher 2015) of creative artifacts, this significantly limits the utility of generative models in the area of computational creativity.

To increase the likelihood of surprise, one could eschew the model distribution and sample from a different distribution or entirely at random. However, this results in outputs that are “frequently more frightening than pleasing”, due to the lack of control over the ”structure / novelty trade-off” (Todd and Werner 1999). Furthermore, a very high level of surprise can be detrimental to the aim of generating outputs that are aesthetically pleasing. This effect is succinctly captured by Wundt’s inverted U-curve, which was later adopted by Berlyne (1971) and others to express the dependency between the hedonic value of a stimulus (e.g. model output) and the novelty of the stimulus as a function that rises to a peak and then falls. Recognizing the necessity of a mechanism to control the generation of surprise while still observing domain-specific structural constraints, Bunescu and Uduehi (2019) introduced a general architecture for two-model sampling, wherein an audience model is trained to learn patterns of expectations, while a composer model learns patterns of surprise, or violations of expectations. By separating the audience learning of expectations from the composer learning of surprise patterns, the proposed Composer-Audience (CA) architecture can generate outputs that confound audience expectations with high probability. This was confirmed in experimental evaluations, where LSTM-based instantiations of the CA architecture were shown to successfully learn patterns of expectation and surprise from distributions of binary sequences.

In this paper, the two-model idea is taken one step closer to computational creativity by using it to learn patterns of expectations and surprise from images of geometric shapes. To this aim, we introduce two GAN-based architectures where the communication of expectations from the audience to the composer is achieved either in the feature space through additive composition, or in the parameter space through norm constraints. Additionally, we establish connections with the domain of style transfer and embed an image-to-image translation module into a third GAN-based CA architecture.
For evaluation purposes, we create benchmark datasets of geometrical figures that capture well defined types of expectations, i.e. empty polygons with 3, 4, and 5 sides, and arrows with various polygons for their heads, and the corresponding surprise, i.e. polygons with one missing side or a grey interior, and arrows with circle or half-circle heads. Experiments demonstrate that the new GAN-based two-model sampling architectures are effective at learning patterns of expectations and surprise in both types of figures. The new GAN-based models are also simpler and more broadly applicable than the style transfer-based instantiations.

**Task Definition and Datasets**

Overall, the task is to train two generative models: an audience $\text{Gen}^a$ that learns a distribution of geometric figures (the expectation), and a composer $\text{Gen}^c$ that learns to confound particular expectations in a well-defined, structured way (the surprise). As detailed in the next section and shown in the generic architecture from Figure 1, both the audience and the composer will be trained within the GAN framework as generators that take as input a vector $z$ of random Gaussian values and produce as output an image denoted as $\text{Gen}^a(z)$ and $\text{Gen}^c(z)$, respectively.

To make the discussion of expectation and surprise more concrete, in Figure 2 we show training samples from each of the three datasets used in the experimental evaluation. The top section shows a CA dataset where the audience is trained to generate polygons with 3, 4, and 5 sides of various relative sizes and orientations. This is the expectation distribution, with samples shown in the row labeled $\text{Gen}^a$. The audience model is trained first, and then kept fixed during the training of the composer model. At training time, the composer has access only to a trained audience producing polygons with 3 and 4 sides, and is trained to generate the same kind of polygons, but with a missing side. This is the surprise distribution, with samples shown in the row labeled $\text{Gen}^c$.

The training is unpaired, e.g. the composer is not given the complete triangle corresponding to the incomplete triangle that it is trained to generate; given that the vertices of the polygons in the training data were generated at random, the corresponding complete triangle is very likely to not even be present in the dataset on which the audience was trained. At test time, the composer is given access to an audience model producing a new type of expectation distribution: polygons with 5 sides. Even though the composer has not seen any pentagons during training, the generalization task is considered successful if the composer can produce the same kind of surprise for pentagons too, i.e. generate the same pentagon as the audience, but with one missing side, as shown in the right section of Figure 3.

The middle section of Figure 2 shows samples from a second dataset where the audience distribution is the same as in the first dataset, i.e. polygons with 3, 4, and 5 sides. However, the composer is tasked with learning to confound the expectation of a white interior by training solely on triangles and quadrilaterals that are colored with random shades of gray. As in the first experiment, the composer is deemed successful if it generalizes to pentagons, a type of polygons that it has never seen at training time, i.e. when used with an audience model that was trained to generate white pentagons, the composer should apply the same surprise pattern and generate gray pentagons.

The bottom section of Figure 2 shows samples from the third dataset, where the audience is trained on a distribution of arrows with 4 types of heads: square, diamond, triangle, and inverted triangle. The arrows can appear at any orientation, with a tail always the same length, whereas the size of the head can vary. The composer confounds the head expectations by replacing the polygon shaped head generated by the audience with either a full circle or half circle head, at random. At training time, the composer sees only square, diamond, and triangle heads, as they are generated by the audience model. At test time, the composer needs to
thwart the audience expectations by replacing the previously unseen inverted triangle heads with full or half circles.

In Figure 3 we show the two kinds of generalization performance that are expected from a CA architecture to be deemed successful. Taking the first dataset as an example, given a random Gaussian vector $z$, the audience model generates a triangle or a quadrilateral and the composer generates the same polygon, but with a missing side, as shown in the first two columns in the figure. Thus, the composer preserves the polygon expectation from the audience, but also surprises the audience by missing one side. We call this within-distribution generalization, because the composer generates surprise for types of polygons (3 and 4 sides) that it has seen at training time. In contrast, the right-hand side of Figure 3 shows examples of out-of-distribution (OOD) generalization. For the first dataset, this requires the composer to generate surprise by missing an edge from pentagons generated (expected) by the audience, in the context where pentagons were never used to train the composer.

**Connections to Other Tasks**  
The CA's generation of surprising features while preserving overall expectations, as illustrated by the within-distribution generalization examples from Figure 3, bears similarities with the mappings performed by unsupervised image-to-image translation and style transfer models, such as CoGAN (Liu and Tuzel 2016), CycleGAN (Zhu et al. 2017), Augmented CycleGAN (Almahairi et al. 2018), or MUNIT (Huang et al. 2018), to name just a few. There are however fundamental differences between learning to surprise in CA models and image-to-image translation. A CA model generates an output image from scratch, e.g. a Gaussian sample, whereas style transfer models use an existing image as input. The CA model’s ultimate objective is to generate truly novel outputs by confounding the expectations of an audience not seen by the composer during training, as shown in the OOD generalization examples from Figure 3, whereas image-to-image translation is traditionally aimed for within-distribution generalization. CycleGAN, for example, documents its limitations on OOD samples and geometric transformations. The requirement that CA architectures learn patterns of surprise that generalize to OOD audiences means they can benefit from domain shift invariance, e.g. Adversarial Discriminative Domain Adaptation (Tzeng et al. 2017), or OOD invariance, e.g. Invariant Risk Minimization (Arjovsky et al. 2020) (but see Rosenfeld et al. 2021 for limitations).

**GAN-based Architectures for Surprise**  
Figure 1 shows the proposed generic GAN-based approach for the CA architecture. The audience model, shown at the top, is instantiated as a prototypical GAN: a standard Gaussian vector $z^a$ is used as input to a Generator network $Gen^a$ that outputs an image $\hat{x}^a$, whereas a Discriminator network $Disc^a$ takes real images $x^a$ and generated images $\hat{x}^a$ as input and is trained to determine whether they are fake or real. The composer GAN, shown at the bottom, has a similar architecture, with one important difference: its Generator network $Gen^c$ uses the expectations computed by the audience generator $Gen^a$. Analogous to the kind of expectations that are engineered by composers in their music, here we use the term expectation in its broader sense to refer to values that are more likely to appear than other values, according to an audience model. These values can refer to the audience model outputs, layers, or even parameters.

There can be many implementations of this generic architecture, depending on how expectations are communicated between the audience and the composer generators. Here, we introduce 3 instantiations:

1. **Layer** based expectations: The composer uses all the layers computed by the audience generator.
2. **Parameter** based expectations: The composer is constrained to be close to the audience parameters.
3. **Output** based expectations: The composer uses the output computed by the audience generator.

In both the layer-based and parameter-based instantiations, the composer generator is set to have the same architecture as the audience generator, i.e. a sequence of transposed convolution layers. The layer-based version is shown in Figure 4, where we chose to communicate expectations by adding at every layer $l_k$ the audience output of the convolution operation to the corresponding convolution output from the composer. The resulting sum is then passed through the usual batch normalization operation, followed by the application of a nonlinear activation function. Note that the addition can theoretically be implemented at any of 3 distinct places, each with a potentially different behavior: before convolution, before normalization, or before activation. The summation is used only in the composer, whereas the audience processes the input as if it were run separately, on its own. Given an input latent vector $z^c$, this means that the composer has access to what the audience would have produced for that input, at every layer. An alternative is to use the summation also as input to the normalization operator in
the audience generator, however this was less effective during evaluations, likely due to the audience parameters being used on distributions unseen during training.

In the parameter-based version, we consider that the audience expectations are captured through its parameters $W_k^a$ at every layer $k$. While the composer’s job is to learn patterns of surprise from data, it is still required to preserve many of the expectations produced by the audience, e.g. it still needs to produce the same kind of shapes as the audience model. We implement this by initializing the composer generator as $W_k^c = W_k^a$ and then fine-tuning the composer while requiring that its parameters do not diverge too much from the audience. This is done by imposing a constraint on the $L_2$ norm of the difference between parameters, i.e., by adding the term $\lambda||W_k^c - W_k^a||^2$ to the GAN objective function.

In the output-based version shown in Figure 5, the composer accesses the audience expectations only as they are produced in its final output image. This is implemented by passing the audience output to an image-to-image translation network that is trained to output the composer distribution.

**Connections to Related Work** In the LSTM-based CA architecture of Bunescu and Ududeci (2019), audience expectations are communicated to the composer LSTM using the output probability distribution of the audience LSTM at each time step. As such, it can be seen as using an Output-based strategy for communicating expectations. Todd and Werner (1999) survey evolutionary approaches to music composition where a composer’s fitness function is guided by judgments that are elicited from a critic every time a new generation of composer models or composer outputs is to be generated. The critic can be a human (unfeasible), rule-based (brittle, fixed aesthetic criteria), learned (from a human critic decisions), or co-evolved with the composer to prefer songs that violate its current expectations. The surprise preference is encoded directly in the fitness function as a difference between the probabilities of the expected and observed notes. With the sole exception of the critic initialization of expectations, which are calculated from a collection of simple folk tune melodies, this evolutionary composer-critic approach generates "musical sequences" entirely from scratch, evolving its own, largely unconstrained aesthetics that "the human user would find worthless". In contrast, our composer-audience approach to generation of surprise is entirely data-driven: structural constraints and expectations are learned from the audience dataset, whereas patterns of surprise reflect the hedonic values implicit in the composer dataset. As such, the CA model learns the aesthetics manifest in the data, be it human or machine generated. Todd and Werner (1999) mention the potential utility of modeling expectation-viion with respect to not only the exposure to previous songs, but also relative to the expectations engendered within the current song. This parallels the long- vs. short-term distinction proposed by Pearce, Conklin, and Wiggins (2004) for the statistical prediction of monophonic music. While modern techniques such as self-attention (Vaswani et al. 2017) may obviate the need for explicit short-term models for prediction, we believe within-output expectations are still important for the task of generating surprise.

**Experiments and Discussion**

We implement the image-to-image component of the Output-based approach using the Augmented CycleGAN. For the audience and composer generators in the Layer-based and Param-based approaches, we use an architecture that starts with 3 transposed convolution layers, followed by 6 ResNet blocks, and ending with 2 more transposed convolution layers. Since a CycleGAN requires training of two generators, the Output-based models require double the number of parameters when compared with the Layer and Param models. LSGAN loss is used during training. To make the audience identifiable between training and testing of the composer, we use a one-hot vector as input to encode the type of shape (e.g., 3, 4, or 5 sides).

As described in the previous section, an audience model is trained first, and then kept fixed while training the composer model, so that the audience expectations do not change. Samples of this separate training of the audience and composer models are shown in Figure 6 for each of the three datasets. On the missing side surprise task, the Layer-based composer obtains the best OOD generalization, as it is able to successfully remove a random edge from pentagons, a type of polygon that it has not seen during training. The Output and Param based composers achieve this OOD generalization with various degrees of success, sometimes removing more than one edge. On the gray interior surprise task, the best OOD generalization is obtained by the Output-based composer, whereas the Layer and Param based approaches have difficulties in preserving the structural expectations, i.e. the actual polygon shape, generated by the corresponding audience model. Finally, on the arrowhead surprise task, all three types of composer models appear to work well, with the Layer-based model slightly edging the other two models in terms of the quality of circle and half-circle heads it generates for both within and out of distribution evaluations.

To determine the importance of keeping the audience model fixed during composer training, in Figure 7 we show samples from experiments where the audience expectations are allowed to change when training the composer. For the Layer-based approach, this means that the audience parameters change when the GAN loss is backpropagated through both the audience generator $Gen^a$ and the composer generator $Gen^c$. The analogue for the Param-based approach was to remove the $L_2$ regularization term $\lambda||W_k^c - W_k^a||^2$ from the GAN objective, which means that the composer parameters are not longer required to be close to the audience parameters. The samples from Figure 7 show that in this...
setting the composer output fails to preserve the structural expectations produced by the corresponding audience. For the polygon datasets, this means that the composer does not follow the polygon shape expected by the audience, whereas for the arrow dataset, the composer does not preserve the tail direction expected by the audience model.

Overall, the experiments with the 3 geometric shapes datasets show that, when the audience and composers are trained separately, the Layer and Output based instantiations of the generic GAN architecture are largely successful at learning patterns of expectation and surprise.

**Acknowledgments**

We would like to thank the anonymous reviewers for their constructive comments and suggestions of related work.

**References**


Rosenfeld, E.; Ravikumar, P. K.; and Risteski, A. 2021. The risks of invariant risk minimization. In ICLR.


9. Poster Papers
When a Computer Cracks a Joke: Automated Generation of Humorous Headlines

Khalid Alnajjar
Faculty of Arts
University of Helsinki
khalid.alnajjar@helsinki.fi

Mika Hämäläinen
Faculty of Arts
University of Helsinki
mika.hamalainen@helsinki.fi

Abstract
Automated news generation has become a major interest for new agencies in the past. Oftentimes headlines for such automatically generated news articles are unimaginative as they have been generated with ready-made templates. We present a computationally creative approach for headline generation that can generate humorous versions of existing headlines. We evaluate our system with human judges and compare the results to human authored humorous titles. The headlines produced by the system are considered funny 36% of the time by human evaluators.

Introduction
Humor, while showcased by a wide spectrum of species in the animal kingdom, has a subcategory that is exclusive to the human being. Verbal humor can only exist in the presence of language, and its generation by computational means is far from trivial.

Humor is effectively a perceiver dependent phenomenon. Nothing can be inherently funny, but humor is perceived and appraised by the mind of a human perceiving it. And ultimately, the perceived humor, if accepted as such by the listener, elicits an emotional response accompanied with a vocal response of laughter that originates from our ancestors, the species before homo sapiens (cf. Ross, Owren, and Zimmermann 2010).

Our paper focuses on generating humor in news headlines. This NLG task is not of the traditional sort, where conveying factual information is the uttermost goal of the system, but rather the affective content of the message is taken into the primary focus of the study.

Automated news generation is a flourishing field with new research being published in a timely manner. This is reflected by the number of recent publications on the topic (Nesterenko 2016; Yao et al. 2017). Quite often, however, as the generated news has to cater for the purpose of communicating facts the question of creativity is set aside. In such a context, there is a trade-off between creativity and communicativity (see Hämäläinen and Honkela 2019).

Whereas creative headline generation is not a new domain to computational creativity, with quite some existing publications on the topic (Lynch 2015; Gatti et al. 2015; Alnajjar, Leppänen, and Toivonen 2019), we aim to intertwine headlines, creativity and humor by proposing a novel method for humor generation that is reasoned by the existing theories on humor.

Our approach alters a word in an existing headline for a humorous effect. We evaluate the method proposed in this paper quantitatively with human judges. We take the different constituents of humor in consideration in the evaluation to uncover the relation of each feature to the humor produced by our system.

Related Work
Humor has received some interest in the past for more than a decade (Ritchie 2005; Hong and Ong 2009; Valitutti et al. 2013; Costa, Oliveira, and Pinto 2015). We dedicate the remaining of this section to describing some of the most recent work conducted on the topic.

Pun generation with a neural model language model is one of the most recent efforts on humor generation (Yu, Tan, and Wan 2018). Their approach consists of training a conditional language model an using a beam search to find sentences that can support two polysemous meanings for a given word. In addition they train a model to highlight the different meanings of the word in the sentence. Unfortunately, they evaluate their system on human evaluators based on three quantitative metrics: fluency, accuracy and readability, none of which tells anything about how funny or apt the puns were.

Alnajjar and Hämäläinen (2018) present a genetic algorithm approach for generating humorous and satirical movie titles out of existing ones. Their method works on a word level replacement and aims for low semantic similarity of the replacement word with the original word to maximize surprise and high similarity with Saudi Arabia to maximize coherence. They consider pun as one of the fitness functions of the genetic algorithm, but the output is not strictly limited to puns. On top the genetic algorithm, they train an RNN model that learns from the genetic algorithm and real people.

Surprise is also one of the key aspects of a recent pun generator (He, Peng, and Liang 2019). They model surprise as conditional probabilities. They introduce a local surprise model to assess the surprise in the immediate context of the pun word and a global surprise to assess the surprise in the
context of the whole text. Their approach retrieves text from a corpus based on an original word - pun word pair. They do a word replacement for local surprise and insert a topic word for global surprise.

An approach building on humor theories is that of Winters, Nys, and De Schreye (2019). The theories are used in feature engineering. They learn templates and metrical schemata from jokes rated by people with a star rating. They embrace more traditional machine learning techniques over neural networks, which has the advantage of a greater interpretability of the models.

Humor has also been tried to recognize automatically in the past. One of such attempts is focuses on extracting humor anchors, i.e. words that can make text humorous, automatically (Yang et al. 2015). A similar humor anchor based approach is also embraced by Cattle and Ma (2018). Both of the approaches rely on feature engineering basing on humor theories. Recently LSTM models have been used for the task of humor detection with a different rates of success (Cai, Li, and Wan 2018; Sane et al. 2019; Zou and Lu 2019).

Humor

Humor is an inherent part of being a human and as such it has provoked the interest of many researchers in the past to formulate a definition for it (see (Krikmann 2006)). Koestler (1964) sees humor as a part of creativity together with discovery and art. In his view, what is characteristic to humor in comparison to the other two constituents of creativity, is that its emotional mood is aggressive in its nature. He calls bisection in humor the collision of two frames of reference in a comic way.

Raskin (1985) presents a theory that is not too far away from the previously described one in the sense that in order for text to be humorous, it has to be compatible with two different scripts. The different scripts have to be somehow in opposition, for example in the sense that one script is a real situation and the other is not real.

In Attardo and Raskin (1991) humor is seen to consist of six hierarchical knowledge resources: language, narrative strategy, target, situation, logical mechanism and script opposition. As in the previous theories, the incongruity of two possible interpretations is seen as an important aspect for humor. An interesting notion that we will take into a closer examination is that of target. According to the authors it is not uncommon for a joke to have a target, such as an important political person or an ethnic group, to be made fun of.

Two requirements have been suggested in the past as components of humor in jokes: surprise and coherence (see (Brownell et al. 1983)). A joke will then consist of a surprising element that will need to be coherent in the context of the joke. This is similar to having two incongruous scripts being simultaneously possible.

Veale (2004) points out that the theories of Raskin (1985) and Attardo and Raskin (1991) entail people to be forced into resolution of humor. He argues that humor should not be seen as resolution of incompatible scripts, but rather as a collaboration, where the listener willingly accepts the humorous interpretation of the joke. Moreover, he argues that while incongruity contributes to humor, it does not alone constitute it.

Generating Humorous Headlines

In their work, Hossain, Krumm, and Gamon (2019) identified several ways people altered news headlines to be humorous. In our method, we aim to model the following ones of their findings: the replacement forms a meaningful n-gram, the replacements are semantically distant, the replacement makes a strong connection with the entity of the headline and belittles an entity or a noun and the replacement creates incongruity. We see the n-gram finding in a broader way of the replacement being compatible with the the existing script (context). The semantic distance is seen as an index of surprise, and the connection between the entity is assimilated with the target of the joke.

The findings we are not focusing on in this paper are that the replacements are sarcastic, suppress tension or have a setup and punchline. The first two are left out as assessing them computationally is a task worth of a paper on their own right, and the third one is left out as it focuses on a particular kind of humor. However, the punchline structure might emerge from the other features being modelled although not explicitly taken into consideration.

In addition to the findings described above, we take the concreteness of the replacement word into account. The reason for this that concrete words are more likely to provoke mental images (see (Burroway 2007)). In fact, we could see this in the humorous training dataset by Hossain, Krumm, and Gamon (2019), where 90% of the most humorous replacement words were concrete as opposed to only 75% of the least humorous replacement words being concrete.

For the above experiment and the rest of the paper, we use the lexicon of 40k common English words that has a concreteness score from 1 to 5 assigned (Brysbaert, Warriner, and Kuperman 2014). If the score assigned with the word is greater or equal to 3, we consider it concrete. The concreteness is evaluated by lemmatizing the word with spaCy (Honnibal and Montani 2017) if it does not exist in the lexicon.

Modelling Humor

Our system operates by taking an existing headline from the corpus of altered headlines (Hossain, Krumm, and Gamon 2019). This corpus has been syntactically parsed by us by using spaCy (Honnibal and Montani 2017), and it has been tagged for the words that should be replaced by its original authors. For a selected headline, our system tries to find suitable humoristic replacement words.

We assess the different potential humorous replacements in terms of multiple parameters, which are prosody, concreteness, semantic similarity of the replacement to the original word and the semantic relatedness of the replacement to negative words describing the target. In this section, we explain how the individual parameters are modelled. An overall view of our method is depicted in Figure 1.

For prosody, we look at the sound similarity between the original word and the replacement. We assess this in
terms of full rhymes, assonance, consonance and alliteration. These are implemented with rules. As the written form of English is notoriously deviant from the phonation, we use eSpeak-ng\textsuperscript{1} to produce IPA transcription for the words the prosody of which is being assessed.

For concreteness we use the values provided in Brysbaert, Warriner, and Kuperman (2014) to score the concreteness of the replacement word. And for semantic similarity we use the pretrained word embeddings from Bojanowski et al. (2017). We use the semantic similarity to assess surprise, in other words, we want to minimize the similarity of the replacement word to the original.

To measure how a new replacement connects to the word selected to be the target of the joke in the headline, a target must first be found. We consider recognized entities in the headline as the potential targets. In case no entities were recognized, we use the subjects in the headline. If neither of them existed, nouns in the headline are treated as target. Out of the list of targets, a random target $t$ is picked to focus on.

For this target $t$, we retrieve words that are related to it to act as descriptive words revealing potential attributes to make fun of. We employ two resources to obtain such knowledge regarding the selected target:

1. The Non-Official Characterization (NOC) list (Veale 2016) which contains information about more than 1000 well-known characters (e.g. Donald Trump and Kim Jung-un) and their expanded stereotypical properties supplied by (Alnajjar et al. 2017) (e.g. Donald Trump: [wealthy, successful, greedy, aggressive, … etc]).

2. A semantic relatedness model built from word associations collected from a web text corpus ukWac\textsuperscript{2}, following the approach described in Meta4meaning (Xiao et al.

\textsuperscript{1}UK English voice. https://github.com/espeak-ng/espeak-ng
\textsuperscript{2}https://wacky.sslmit.unibo.it/doku.php?id=corpora

We chose to base our relatedness model on a web-based corpus instead of a news-based one to favor discovering related words from various domains, which would be perceived as more humorous.

If the target $t$ is an entity, we search the first resources (i.e. the NOC list and the expanded properties) to collect its top $k$ stereotypical properties. In case no available knowledge regarding the entity existed, we attempt to acquire the top $k$ related words to the rest of the potential targets (subjects and nouns, respectively) using the second resource (i.e. the semantic relatedness model). In our case, we empirically set $k$ to 100 to allow diversity and reduce noisy relations, while ensuring the descriptiveness of the words to the target.

To be able to place the target in a humorous light, we only regard the descriptive words that describe it negatively, which is determined by employing a polarity classifier provided by Akbik, Blythe, and Vollgraf (2018). Lastly, the connection of the replacement word to the target is assessed based on the semantic relatedness between the replacement word and the target’s negative descriptions. We desire to maximize such connections to encourage replacements that are associated with the target from a negative angle.

**Generation and picking out the best candidate**

We use the Humicroedit dataset of headlines published by Hossain, Krumm, and Gamon (2019) as the source of original headlines. Furthermore, the dataset contains edits performed by humans to make the headlines humorous along with a score indicating how humorous they were when perceived by other people on a scale from 0 to 3. The motivation for using this dataset is that the editors were required to make a single change to either a verb or a noun in the head-
line to make it humorous, which focuses the scope when modeling such a process computationally.

In our generation method, we only consider headlines where the original word that is selected to be replaced is parsed as either a noun or a verb using spaCy and is a single token (i.e. ignoring cases such as “Illegal Immigrants”).

The rational behind is to reduce misparsing errors and concentrate on a single-word changes.

For an original headline $h_o$ with its selected word to be replace $w_o$, our method converts it into a humorous one $h_t$ by replacing $w_o$ with another word $w_p$ as follows. It begins by acquiring replacement candidates $C$ that fit the syntactical position of the selected word $w_o$ by querying a massive syntactical repository of grammatical relations that have a frequency greater than 50 in a web-based corpus (Alnajjar 2018) (see Figure 2 for an example of a grammatical relation in the repository). By considering candidates that are apt to the existing syntactical relations in the headline, we ensure that the new replacement has syntactic cohesion and suits the grammatical context.

To illustrate how the method works, let’s consider the headline $h_o = “City$ halls and landmarks turn green in support of Paris climate deal” as an example, where the word to replace $w_o = climate$.

After parsing this headline, we find that the to-be-replaced word $w_o$ is a noun (NN) and has a dependency (compound) on the word deal (NN). We query the syntactical repertory to find potential replacements that suit this relation, which yields 58 candidate replacements ($C = \{ ‘loan’, ‘business’, ‘cash’, ‘oil’, ‘holiday’, ‘peace’, ‘content’, ‘drug’ ...\}$).

In the next phase, the method removes the original word $w_o$ from the candidates if it existed and prunes out any candidate word in $C$ that is not identified as concrete (i.e. having a concreteness score greater or equal to 3 based on (Brysbaert, Warriner, and Kuperman 2014)). As a result, candidate words such as ‘peace’ and ‘content’ in the earlier example are removed resulting in a total of 34 candidates.

If there is more than 500 replacement candidates (e.g. in situations where the token to replace is a verb and is the root of the phrase), we randomly select 500 candidates in $C$ to be examined. This is performed to reduce the search space that the method will traverse and to efficiently discover local optimal solutions as there is no particular global optimal solution for the task we are addressing.

Replacement candidates are then evaluated on the four humor aspects we are modeling, which are 1) prosody, 2) concreteness score, 3) inverted (i.e. minimized) semantic similarity between the original word $w_o$ and the candidate $c$, and 4) the semantic relation between the candidate $c$ and the negative words of the selected target $t$. As we are dealing with multiple criteria for modeling humour, we adopt a non-dominant multi-objective sorting approach (Deb et al. 2000) to find and select candidates in the Pareto front. Additionally, applying a non-dominant sorting for creative tasks (e.g. generating humour) increases the chances of finding balanced and diverse solutions that are more likely to be deemed good (Alnajjar, Hadayutullah, and Toivonen 2018).

Applying the evaluation and the non-dominant sorting on the example headline, the method highlights candidates such as ‘cash’, ‘meal’, ‘drug’ to be chosen as replacements. For the same example, the original word climate was replaced with marijuana by a human editor in the Humicroedit dataset. Interestingly, marijuana is a drug and our method was able to suggest it.

## Results and Evaluation

To evaluate our method, we randomly select 83 headlines from the Humicroedit dataset that meet our criteria specified earlier. For each headline, we request our method to produce humorous alternatives, ranked by the non-dominant sorting, out of which we randomly select 3 to be evaluated from the top humorous headlines.

Table 1 shows some of the headlines generated by our approach. The humorous replacement word is marked in bold. The original word and the replacement word suggested by a human from the corpus are shown in their respective columns.

We conduct our evaluation on Figure-Eight\(^3\), which is a crowd-sourcing platform that assigns paid reviewers for tasks such as questionnaires. We evaluate all the 3 variations produced by our system for the 83 headlines, showing the original headline as well. In addition, we evaluate the human edits for the same headlines from the dataset. The reviewers were not told they were evaluating computer generated humor, as the mere knowledge of a computer being an author of a creative artefact is known to provoke a bias towards seeing the generated output in a more negative light (see (Colton, Wiggins, and others 2012)).

We asked five people to rate the headlines based on the following questions:

1. The altered headline is humorous.
2. The altered word is surprising.
3. The altered word fits into the headline.
4. The altered word is concrete.
5. The joke of the headline makes fun of a person or a group of people (also known as the target of the joke).
6. The altered word shows the target in a negative light.
7. The altered word is a pun of the original word.

We evaluate the first two questions on the scale of 0 to 3 (Not funny, Slightly funny, Moderately funny and Funny). Or surprising in the case of the Q2) similarly to the questions for humor in Hossain et al. (2017). The rest of the

\[^3\]https://www.figure-eight.com/
<table>
<thead>
<tr>
<th>Humorous headline by our system</th>
<th>Original word</th>
<th>Human replacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thieves carry out elaborate van heist to steal millions in cereal, Swiss police say</td>
<td>cash</td>
<td>blouses</td>
</tr>
<tr>
<td>Trump eats the wrong Lee Greenwood on Twitter</td>
<td>tags</td>
<td>woos</td>
</tr>
<tr>
<td>‘I was very angry’ at Trump, says Myeshia Johnson, widow of fallen sock</td>
<td>soldier</td>
<td>cake</td>
</tr>
<tr>
<td>Trump Tried To Climb Heather Heyer’s Mother During Funeral: ‘I Have Not And Now I Will Not’ Talk To Him</td>
<td>call</td>
<td>proposition</td>
</tr>
<tr>
<td>U.S. says Turkey is helping ISIS by Combing Kurds in Syria</td>
<td>bombing</td>
<td>feeding</td>
</tr>
</tbody>
</table>

Table 1: Examples of generated headlines.

questions are presented as yes/no questions. The sixth question is only visible if the fifth question has been answered to affirmatively.

Figure 3: Percentages for each evaluation question. H marking human authored headlines, and S computer authored ones.

Figure 3 shows the percentages of the result for each question from the human evaluation. The results for the human edited titles (H) and the ones produced by our method (S) are shown side by side. From the question 3 onward, 0 marks negative and 1 affirmative answer. All in all, our system scored slightly lower on the questions than real people, which is to be expected due to the difficulty of the problem. However, our system got slightly better results in the question number 6, which means, that when the system had originally picked a target, it managed to convey negativity towards the target on a level comparable to a real human.

In terms of humor, our system managed to produce at least slightly humorous headlines 36% of the time, whereas people produced at least slightly humorous headlines 56% of the time. In comparison, for a recent pun generator, (He, Peng, and Liang 2019) report a success rate of 31% for their system according to a human evaluation, to put our results in a computational perspective.

Table 2 shows the results form another perspective. The score row shows the results for human authored titles in the original publication (Hossain et al. 2017), whereas the human row shows the results for the very same titles in our evaluation. The max shows the average of the best scoring generated headline out of the 3 ones produced for each original headline, and min shows the average of the worst headline in the triplets. Avg is the average of the scores for all the generated headlines.

By looking at the results this way, we can see that at best, our method can produce humor comparable to real humans in the scale of funniness, with a higher amount of surprise, better aptness of the replacement word to the context, higher level of concreteness, higher negativity towards the target and higher level of punniness, falling shorter only in the case of having a perceivable target for the joke in the headline. Focusing on the best scoring individuals might sound like giving too good a picture of the performance of the system, however, they set the upper boundary for the performance of the system. This being said, with the exact same method, better results could be obtained in the future by developing a better way for ranking the humorous headline candidates output by the system.

By considering the headlines produced by our method that have the maximum score for an original headline, we see that 47 of them were credited as humorous (i.e. having a score ≥ 1) out for the 83 original title. On the other hand, 43 of the human generated were considered humorous.

In the following analysis, we aim to evaluate the different criteria considered in our method for modeling humour. In terms of prosody, we look at the number of times a headline was considered to be punny by people with respect to our method’s score on the prosody dimension. Overall, 22% of the generated headlines were considered to have a pun in relation to the original word. Out of these headlines, 88% of them were evaluated positively on the prosody dimension by our system. This indicates that the method exhibited capability of assessing the sound similarity and punniness to the original word.

For the concreteness, we are considering concrete words defined in (Brysbaert, Warriner, and Kuperman 2014) as candidates. As a result, we expected to have headlines produced by the method score high on the fourth question. Contrary, only 56% of them were deemed concrete. This indicates that a more robust model is required to model the concreteness of terms.

By observing Figure 3, we notice that 46% of headlines suggested by our method are considered surprising (i.e. scoring at 1, on average). As we are using a word embeddings model, it is difficult to come up with a semantic
Table 2: Mean and standard deviation of altered headlines by humans and our method.

<table>
<thead>
<tr>
<th></th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>µx</td>
<td>SD</td>
<td>µx</td>
<td>SD</td>
<td>µx</td>
<td>SD</td>
<td>µx</td>
</tr>
<tr>
<td>score</td>
<td>0.99</td>
<td>0.62</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>human</td>
<td>0.97</td>
<td>0.49</td>
<td>0.89</td>
<td>0.41</td>
<td>0.58</td>
<td>0.23</td>
<td>0.6</td>
</tr>
<tr>
<td>max</td>
<td>0.97</td>
<td>0.45</td>
<td>0.98</td>
<td>0.36</td>
<td>0.69</td>
<td>0.19</td>
<td>0.7</td>
</tr>
<tr>
<td>avg</td>
<td>0.6</td>
<td>0.28</td>
<td>0.67</td>
<td>0.23</td>
<td>0.53</td>
<td>0.15</td>
<td>0.56</td>
</tr>
<tr>
<td>min</td>
<td>0.28</td>
<td>0.27</td>
<td>0.37</td>
<td>0.24</td>
<td>0.36</td>
<td>0.18</td>
<td>0.4</td>
</tr>
</tbody>
</table>

similarity threshold that separates similar words from non-similar ones, especially for modeling surprisingness. Therefore, we test the scores assigned by the models on three thresholds of similarity (0.3, 0.2 and 0.1) with respect to the headlines viewed as surprising by online people. Out of the 46% surprising headlines, 98%, 84% and 40% headlines are considered to be dissimilar by the semantic model by using the three above mentioned thresholds. This indicates that minimizing the semantic similarity increases surprise to a degree, after which lowering the similarity results in a lower surprise.

Lastly, we perform the same analysis regarding the connection between the replacement word and the selected target with respect to question five and six. 75% of the time, our function scored positively on headlines evaluated as making fun of a target. Out of which, 77% were correctly seen as negative by the method with respect to Q6.

### Discussion

As the best headlines produced by our system for each original headline can, on the average, reach to a human level in terms of most of the factors measured by our evaluation, an immediate future direction for our research is to develop a better ranking mechanism to reach to the maximum capacity of our system. Perhaps such ranking could be learned by training an LSTM classifier on humor annotated corpora such as the one used in this paper or the one proposed by (West and Horvitz 2019).

For surprise, we opted for a rather modest approach by assimilating it to an inverse semantic similarity to the original word. However, different metrics have been proposed to model this phenomenon, such as a neural network based composer-audience model (Bunescu and Udeuehi 2019) or probabilistically modelling the likelihood of a certain word occurring in a given context (see (Degaetano-Ortlieb and Piper 2019)).

The particularly low score on the concreteness highlights the inadequacy of using an annotated lexicon for its assessment. Perhaps, in the future, concreteness could be modelled in a more robust context dependent way. Previous work (Naumann, Frassinelli, and im Walde 2018) exists showing differences in the distributional representations of concrete and abstract words. As word embedding models are based on the distributional hypothesis, this discovery could be exploited for a context dependent classification by using context-aware word embeddings.

If the method was to be used as a tool for assisting journalists when composing news articles, the fact that employing computational methods for headline generation might result in offensive headlines (see (Alnajjar, Leppänen, and Toivonen 2019)) has to be taken into account. Our humor model maximizes the negative relation to its target, which might be considered as an insult, if understood in a wrong, non-humorous fashion.

Our current approach focuses on English, in the future, we are interested in using our method for other languages as well such as Finnish. This would require a more robust surface realization method to deal with morphology more complex than that of English (Hämäläinen and Rueter 2018). There is already a similar semantic database available for Finnish (Hämäläinen 2018) as the one we used for English, which greatly facilitates a multilingual port of our method.

### Conclusions

We have presented a method for generating humorous headlines that in its current state, falls behind the human level humor. Nevertheless the results reach to a comparable level with an existing neural based method. The method proposed by us has the potential of reaching to a human level humor generation in the limited domain task of altering a word in an existing headline if a better ranking mechanism for its output was introduced.

The evaluation and analysis we conducted on the results has revealed several features which can be modelled better in the future to improve our method. As we have gathered human judgements for headlines generated by our system for original headlines that are based on an existing humor annotated corpus, we are releasing our evaluation results and the generated titles⁴ in the same format as the corpus we used so that our data can be easily used in research dealing with the existing dataset.

### Acknowledgments

This work has been partially supported by the European Union’s Horizon 2020 programme under grant 825153 (Embeddia).

### References

27th International Conference on Computational Linguistics, 1638–1649.


Costa, D.; Oliveira, H. G.; and Pinto, A. M. 2015. In reality there are as many religions as there are papers-first steps towards the generation of internet memes. In ICCC, 300–307.


Winters, T.; Nys, V.; and De Schreye, D. 2019. Towards a General Framework for Humor Generation from Rated Ex-
Ukiyo-e Analysis and Creativity with Attribute and Geometry Annotation

Yingtao Tian
Google Brain
Tokyo, Japan

Tarin Clanuwat
ROIS-DS Center for
Open Data in the Humanities
NII

Chikahiko Suzuki
ROIS-DS Center for
Open Data in the Humanities
NII

Asanobu Kitamoto
ROIS-DS Center for
Open Data in the Humanities
NII

Abstract
The study of Ukiyo-e, an important genre of pre-modern Japanese art, focuses on the object and style like other artwork researches. Such study has benefited from the renewed interest by the machine learning community in culturally important topics, leading to interdisciplinary works including collections of images, quantitative approaches, and machine learning-based creativities. They, however, have several drawbacks, and it remains challenging to integrate these works into a comprehensive view. To bridge this gap, we propose a holistic approach: We first present a large-scale Ukiyo-e dataset with coherent semantic labels and geometric annotations, then show its value in a quantitative study of Ukiyo-e paintings’ object using these labels and annotations. We further demonstrate the machine learning methods could help style study through soft color decomposition of Ukiyo-e, and finally provides joint insights into object and style by composing sketches and colors using colorization. We make our dataset (Tian, CODH, and ARC 2021) available online 1.

Introduction
The Edo period of Japan (16th to 19th century) has seen the prosper of Ukiyo-e (浮世絵), a genre of pre-modern Japanese artwork that consists of paintings and woodblock printings. Unlike early dominating Emakimono (絵巻物, picture scroll) and Ehon (絵本, picture book) that focus on famous figures and stories in Sinosphere culture and classic Japanese stories, the topic of Ukiyo-e extends broadly to daily subjects, such as characters like beauties and Kabuki (歌舞伎), landscape arts, animals and plants in everyday life, and even contemporary news. As an example, Figure 1 shows an Ukiyo-e depicting a Kabuki performance. The popularity of woodblock printing makes it possible to produce paintings on a larger scale at a lower cost, which contributes to the flourish of Ukiyo-e and leaves us with a vast collection of artworks in this genre (Kobayashi 1994; IUS 2008). Such an extensive and varied collection provides a valuable corpus for Japanese artwork research.

The subject of such artwork study could be multi-faceted involving several aspects, of which two crucial are the object in the painting, such as the outline and the shape of depicted figures, and the style of painting, such as textures and colors. For example, the former reveals the trend of objects depicted over time, and the latter allows the identification of artists (Suzuki, Takagishi, and Kitamoto 2018). The renewed interest by the machine learning community in the culturally essential topics has led to works addressing the traditional Japanese artworks from an interdisciplinary perspective. Along this line of research, building open collections of digitized images has been proposed for Ehon (Suzuki, Takagishi, and Kitamoto 2018) and Ukiyo-e (Art Research Center, Ritsumeikan University 2020; Pinkney 2020). Further works use quantitative approaches into the object for artworks, such as studying the geometry features of Buddha statues (Renoust et al. 2019) and Ukiyo-e faces (Renoust et al. 2019). Alternatively, inspired by the art nature of painting, machine learning-based creativity has been leveraged for studying style, such as painting process generation (Tian et al. 2020) and image synthesis across artwork and photorealistic domains (Pinkney and Adler 2020). These works provide valuable connections between machine learning and the humanities research of Japanese artwork.

We, however, also notice that these works present several drawbacks. For example, collection on digitized images may either comes with no semantic (Pinkney 2020) or is in a format not designed with machine learning-based applications in mind. Furthermore, quantitative approaches are only conducted on a small set of artworks (Murakami and

11https://github.com/rois-codh/arc-ukiyoe-faces

Figure 1: An example of Ukiyo-e work in ARC Ukiyo-e Collection (Object Number arcUP2452) titled Kizukansuke by painter Hirosada. The painting on the left is accompanied by metadata for this work on the right. For example, metadata further indicates this work is a middle-sized Nishiki-e (multi-colored woodblock printing) produced in 1849.
Urabe 2007) or require extensive human labor to adapt for Ukiyo-e (Renoust et al. 2019), and machine learning-based creativity works may deal more with cross-domain art expression (Pinkney and Adler 2020) than the very domain of artwork on which humanities research focuses. Finally, the art study into a particular genre requires insights into both the object and style to acquire a comprehensive understanding. Current works, however, only address one of the object or style, falling short of the expectation.

To overcome the aforementioned drawbacks and to provide deeper insight into the artistic style of Ukiyo-e, we propose a new approach that is (1) holistic in both studying the object and style through the joint use of images, labels, and annotations, and (2) powered by large scale data and state-of-the-art machine learning model than the prior works. To summarize, our main contributions are as follow:

- We present a large-scale (11,000 paintings and 23,000 faces) Ukiyo-e dataset with coherent semantic labels and geometric annotations, through augmenting and organizing existing datasets with automatic detection.

- We are the first to conduct a large-scale quantitative study of Ukiyo-e paintings (on more than 11,000 paintings), providing understanding into object in artworks by jointly quantifying semantic labels and geometric annotations.

- We show that machine learning-based models could provide insights into style by decomposing finished Ukiyo-e images into color-split woodblocks that reflect how Ukiyo-e images were possibly produced.

- We study and show machine learning-based creativity model could engage problems that arise jointly studying object and style by separating geometry shapes and artistic styles in an orthogonal and re-assemblable way.

### Dataset

Art research in traditional paintings often asks questions regarding the work, like the author and production year. One focus in such research is on faces since they could help answer these questions through quantitative analysis. In this direction, Collection of Facial Expressions (Suzuki, Takagish, and Kitamoto 2018; Tian et al. 2020) provides a large-scale (8848 images) set of coarse-grained cropped faces. Another study (Murakami and Urabe 2007) deals with facial landmarks which are more fine-grained than cropped faces to support quantitative analysis. However, its manual labeling process only allows analysis on a small set (around 50 images) of Ukiyo-e paintings.

To combine both works’ advantage, we extend existing datasets through augmentation and automated annotation, resulting in a large-scale Ukiyo-e dataset with a more fine-grained facial feature. The rest of this section details the process and analysis of our new proposed dataset.

### Fundamental Datasets

We build our work based on two foundation datasets. One of them is ARC Ukiyo-e Collection (Art Research Center, Ritsumeikan University 2020), a publicly available service that provides access to digitized Ukiyo-e paintings primarily in the Edo period, plus metadata compiled by domain experts. It has 11,103 entries of painting and the associated metadata, one example of which is shown in Figure 1. This service allows researchers to dive into curated metadata for comparative study for art research.

Another dataset is Ukiyo-e Faces Dataset (Pinkney 2020), a public available dataset of Ukiyo-e faces extracted from Ukiyo-e images available online. With 5,000 high-quality faces, this dataset plays an essential role in controllable image generation across Ukiyo-e faces and photo-realistic human faces (Pinkney and Adler 2020). However, as this dataset focuses on image synthesis, it does not include metadata for Ukiyo-e paintings from which faces are extracted.

### Geometric Annotation with Facial Landmark Detection

Inspired by Pinkney (2020), we use an face recognition API, Amazon Rekognition (link), to detect facial landmarks in in Ukiyo-e Faces Dataset paintings. Despite targeting photo-realistic human face images, this API demonstrates compelling accuracy on Ukiyo-e paintings. Since the detected faces may not be well-aligned, we infer the possibly rotated bounding box of faces for cropping faces from the

<table>
<thead>
<tr>
<th>Facial Region</th>
<th>Landmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left Eye</td>
<td>Center, Left, Right, Up, Down</td>
</tr>
<tr>
<td>Right Eye</td>
<td>Center, Left, Right, Up, Down</td>
</tr>
<tr>
<td>Left Eyebrow</td>
<td>Left, Right, Up</td>
</tr>
<tr>
<td>Right Eyebrow</td>
<td>Left, Right, Up</td>
</tr>
<tr>
<td>Left Pupil</td>
<td>Center</td>
</tr>
<tr>
<td>Right Pupil</td>
<td>Center</td>
</tr>
<tr>
<td>Mouth</td>
<td>Left, Right, Up, Down</td>
</tr>
<tr>
<td>Nose</td>
<td>Center, Left, Right</td>
</tr>
<tr>
<td>Jawline</td>
<td>Upper Left &amp; Right, Mid Left &amp; Right, Chin Bottom</td>
</tr>
</tbody>
</table>

Figure 2: An example of detected landmarks and the extracted face in Figure 1’s Ukiyo-e painting. On the left, the red dots show detected facial landmarks and the rectangle shows the bounding box inferred from these landmarks. The right image shows the extracted face from the bounding box. The table lists a summary of all landmark locations.

Figure 3: Faces with their landmarks. In each row, we show six examples of extracted faces annotated with their corresponding landmarks in the same format as Figure 2.
painting, inspired by the preprocessing in FFHQ (Karras, Laine, and Aila 2019). In Figure 2 we show an example of detected landmarks and the face extraction process.

A total of 18,921 faces and their corresponding facial landmarks have been detected from paintings in ARC Ukiyo-e Collection. Furthermore, since Ukiyo-e Faces Dataset (Pinkney 2020) also follows the same preprocessing as FFHQ, its 5,000 faces are comparable to the faces extracted from ARC Ukiyo-e Collection. Although faces in Ukiyo-e Faces Dataset lack metadata, we can still incorporate them for geometry statistics by going through the above-mentioned landmark detecting process. In doing so, we have a total of around 23,000 Ukiyo-e faces. In Figure 3, we show examples of such faces and their landmarks.

Semantic Labels Incorporation

As our dataset is derived from ARC Ukiyo-e Collection, we can also relate faces and the corresponding landmarks with the original metadata, such as the year of creation and the author of the painting. In Figure 4 we show these two metadata jointly, as well as exemplary paintings belonging to several authors. For example, we can observe Shumei (襲名, name succession) system common in traditional Japanese art community where an artist takes his/her teacher’s name, as the case of the lineage of Kunisada 1st gen (国貞 初代), Kunisada 2nd gen (国貞 二代目) and Kunisada 3rd gen (国貞 三代目). Furthermore, we can also notice three peaks of production of Ukiyo-e painting, occupying the early, mid, and late 19th century. The last peak is dominated by Kogyo (耕漁) who painted well into the 20th century and whose
Figure 5: Study of landmark quality by comparing automatically detected positions (a) with expert labeled positions (b). As Ukiyo-e faces are mostly towards either left or right, we normalize all paintings to face left for the geometry purpose. The study has been conducted for 69 Ukiyo-e paintings, and the mean error of pixel distances are aggregated in (d), which we use to decide which landmarks are considered high-quality. The decision is based on picking landmarks with a low error of pixel distance (heuristically those < 20), except for Jawline (*) that needs special consideration: as (c) shows, landmarks on the the direction of facing (Upper Left, Mid Left) are useless since they are invisible in most Ukiyo-e. The others (Upper Right, Mid Right, Chin Bottom) are valuable since they still lie on the jawline, and they are far from other landmarks, allowing larger error merging when used for calculating angular features.

uniqueness is further shown in his exemplary paintings under the influence of modern painting.

Experiment

Study Ukiyo-e Object using Geometry Features

Regarding the content, art researches may be divided into two categories: the shape that deals with geometry features and the texture that deals with brushes and color features. To quantitatively provide insights on attributes such as the author and the painting year, either category can be used for unsupervised learning, like clustering, or supervised learning, like predicting metadata. While the texture features could help analyze attributes for a single work (Tian et al. 2020), the geometry features could also be considered since the texture may vary due to Ukiyo-e’s frequent reprint (Murakami and Urabe 2007) or sculpture’s preservation condition (Renoust et al. 2019). Both works propose to leverage

facial landmarks to infer geometry features such as angles and iconometric proportions to quantify artwork.

However, since both works rely on manually labeled landmarks, they either suffer from being too small (only around 50 Ukiyo-e paintings are annotated with landmarks) or require extensive human effort if we ever want to apply the technique used on sculpture to Ukiyo-e. To bridge this gap, we propose to use automatically detected landmarks as geometry features. To our best knowledge, we are the first to conduct large-scale (more than 10k paintings) quantitative analysis of Ukiyo-e painting. We hope it could serve as an invitation for further quantitative study in artworks.

Geometry Features from Landmarks Inspired by Murakami and Urabe (2007), we consider the angles formed by landmarks as they are geometry-invariant under rotation. Here we focus on rotation in two-dimensional space. As traditional Japanese painting are not photo-realistic, the 3D perspective in viewing is a more complex issue involving deformation and we left it for future study. To attain a clear understanding of the quality of landmarks, we conduct a study on 69 Ukiyo-e paintings, comparing landmarks that are automatically detected with positions manually annotated by domain experts, as detailed in Figure 5. We observe that, despite the general high-quality of the predicted landmarks on Ukiyo-e painting, some landmarks have systematically worse quality than others we decided not to consider. In the end, we calculate 252 angles formed by all possible triplets of high-quality landmarks as geometry features for each face, as illustrated in Figure 6.

Analysis on Authorship To illustrate the information of geometry features, we conduct unsupervised (PCA, T-SNE, UMAP) and supervised (LDA) clustering of faces using geometry features in Figure 7. All clusterings show two distinctive authors, Kogyo and Hirosada (貞広, 弘右), are separated from other authors. Such separation could be supported through visual inspection into original paintings. For example, Figure 4 (c) shows Kogyo and Hirosada has visually distinctive styles compared to other painters. Furthermore, such separation could also be cross-verified with analysis leveraging other information sources, Figure 4 (a) shows that Kogyo was active well into the 20th century where in contrast Ukiyo-e paintings are mainly around the middle 19th century. Furthermore, his uniqueness of style is visu-
Comparing Ukiyo-e, Ehon and Human Faces  As exemplary paintings in Figure 4 (c) show, Ukiyo-e paintings are characterized by their particular facial geometry, which could potentially be different from other art genres or photo-realistic human faces. To quantify such observation, we conduct unsupervised (PCA, T-SNE, UMAP) clustering of Ukiyo-e (popular in the 19th century) faces, Ehon / Emakimon (another Japanese artworks genre popular in the 16th to 17th century) faces, and realistic human faces.

Concretely, we use Kaokore (Tian et al. 2020) for Ehon / Emakimo faces, as well as human face photos collected in FFHQ (Karras, Laine, and Aila 2019) dataset that are published under CC BY 2.0 license. In Figure 8, we can observe that the geometry of Ukiyo-e faces is different from Kaokore, and only share similarities to a small section of realistic human faces. This observation confirms the uniqueness of Japanese artworks’ way of portraying humans compared to the real-world image and shows that the development of Japanese artworks over time is a drastic one.

Study Ukiyo-e Style through Color Separation  Ukiyo-e printings distinguish themselves from other traditional Japanese artworks by the very manner of producing. Unlike Ehon, which is targeted at a small audience and thus painted by hand, Ukiyo-e is mass-produced using woodblock printing after the painter finishes the master version. As shown in a modern reproducing process (link), multiple woodblocks are carved, each for a portion in the image with a single color, and are printed sequentially with corresponding inks onto the final canvas. Unfortunately, such a process for a given Ukiyo-e painting is not precisely reproducible since the underlying woodblocks are vulnerable, easily worn-out, and often discarded after a certain number of prints. Thus from an art research point of view, it would be interesting to recover the above-mentioned separated portions for a given Ukiyo-e painting with only access to the image itself.

We address this challenge by framing it as a soft color segmentation (Aksoy et al. 2017) task, which decomposes an input image into several RGBA layers of homogeneous colors. The alpha channel (“A” in “RGBA”) in each layer allows pixels to potentially belong to multiple layers, which captures ambiguity unavoidable due to imperfect woodblock carving and alignment in multi-pass printing. In detail, we use state-of-the-art Fast Soft Color Separation (FSCS) (Akimoto et al. 2020) for efficient processing. As shown in Figure 9, FSCS decomposes Ukiyo-e paintings into layers of homogeneous colors using color palette. The inferred layers could be interpreted as woodblocks with corresponding colors that could be used for making a particular artwork.

The decomposition of a painting into multiple layers of homogeneous colors allows us to explore further creativity. One example in this direction is recoloring, where we pick a new color for each of the individual layers and compose them into a recolored painting. As shown in Figure 10, the recoloring could be done either automatically using the inferred color palette from other artworks or manually in Adobe After Effects with alpha add mode for blending. The recoloring here serves as an example to study artworks and opens the door to reinterpret them in a new way.
Figure 9: Soft color separation takes as input Ukiyo-e paintings (left) and a color palette (middle), and produces decomposed layers of homogeneous colors (middle). These layers can be used as the inferred woodblocks for corresponding colors and composed back to a reassembled painting (right) resembling the original one. We infer the color palette by applying K-means clustering (Lloyd 1982) on the input painting’s pixels.

Figure 10: Decomposing an Ukiyo-e painting (a) with color palette (b) and recoloring, which could be done automatically (c, d) using color palettes inferred from the reference images, or manually with Adobe After Effects (e).

Study Jointly Ukiyo-e Object and Style by Composing Sketch and Color

As we deal with a dataset focusing on artworks, it becomes natural to ask whether we could engage them with approaches invoking creativity and artistic expression. One direction is to examine whether the recent advances of machine learning models could create structurally sound, or even artistically impressive, results. In this direction, generative models have been proposed to generate faces in Japanese painting style (Tian et al. 2020) and blend generative models trained on data of different domains by swapping layers of two image generation neural networks (Pinkney and Adler 2020). However, the former lacks controllability in the generation as it can only produce images as a whole, and the latter focuses on transferring across separated, different domains by the nature of its design.

Thus we identify an unbridged gap in the in-domain separation of artistically essential aspects. In detail, we ask the following question: what is the (dis)entanglement between the object and style within the Ukiyo-e? Answering this question reveals the relation between Ukiyo-e’s object and style. Furthermore, it also allows editing one of them while keeping another intact for creative expression. One way to separate the object and style is to represent the former with line art sketches for what person/scene is depicted, and the latter with color and texture information showing the painting style. They could be composed with a colorization process, which blends a sketch as an object reference and an image as a reference for instance-level painting style.

Face Images. We extract line art sketches from Ukiyo-e
Figure 13: Matrices of blending line art sketches and painting style for Ukiyo-e faces (a) and whole Ukiyo-e paintings (b). Within a single matrix, each row represents an art line sketch, each column represents the reference image for style, and images at an intersection are the blending results of the corresponding row and column.

The model learns to separate the object, indicated in the sketch image, and the style, indicated by reference image, as two orthogonal and composable semantics, it could blend arbitrary combination of sketch and reference style images. Such separation could enable future works to help with humanities research on combinations of Ukiyo-e color and subject. For example, in Ukiyo-e depicting Kabuki, the attributes and colors of the characters are somewhat correlated semantically. Therefore swapping colors can change the meaning of scenes and people in the painting. We envision that discoveries could be made by studying how the impression of Ukiyo-e paintings changes through the process of swapping colors.

Whole Painting. We go beyond faces and work on whole Ukiyo-e painting images. By employing the same pipeline...
to the whole painting images, as shown in Figure 14, the
to the whole painting images, as shown in Figure 14, the
model can be further leveraged to colorize in-the-wild wood-
block printing images, as Figure 15 shows. However, while
the resulting colorized images are reasonable, they are of
lower quality than those of faces. Such observation is
anticipated since the whole Ukiyo-e painting is more complex
than face in many ways, like topics and topological config-
uration of objects, which presents a much more challenging
task for colorization. This issue could be further exaggerated
by the discrepancy between the Ukiyo-e domain where the
model is trained and the woodblock painting domain where
the model is applied. We would leave higher quality, whole
Ukiyo-e painting colorization for future study.

Conditional vs. Unconditional Colorization. While we
choose to use a conditional colorization method, which pro-
duces results from a sketch and a reference image for color
and styles, it is also worth considering a simper, uncondi-
tional colorization method that directly generates the results
from a sketch, such as Pix2PixHD (Wang et al. 2018). This
alternation, however, suffers from the inability to control the
color and style of the generated image. Moreover, as we
show in Figure 16, the unconditional colorization method
produces worse colorization results than the conditional
colorization method (Lee et al. 2020). We argue that this is
expected since the former method has to fall back to safe
colors that valid for any Ukiyo-e images, while the latter
could make a wiser choice based on the reference images.

Discussion We show that Ukiyo-e paintings can be studied by (1) representing object with line art sketches, (2) repre-
senting style as a color reference image, and (3) composing
them using colorization. This pipeline provides a clear sep-
oration of two semantics important in the art research and
allows further creativity through compositions of both in un-
seen ways. As it is just one possible way of studying the in-
teraction between the object and the style, we expect further
works could explore different forms of creative expression.

For example, one possible further work on computational
creativity could be focused on the controlled generation of
Ukiyo-e images. Although we proposed to using sketch and
color as object and style for image composition, they
nonetheless could take other forms, such as categorical vari-
ables like a person’s social status, gender, or the painter’s
style factors such as the art school. Furthermore, another
research direction could be on interpretability in a cultural
sense, where the association between styles and color back-
ground can be revealed by exploring the generated images’
factors. Finally, we also envision that creative work can
jointly consider the information in multiple modalities. In
this direction, one may consider the relation of painters as
desecrate graph with the paintings themselves as continu-
ous images, combining graph analysis such as Graph Neural
Networks (Scarselli et al. 2008) and image processing tech-
niques.

Conclusion

In this work, we propose to bridge the machine learning and
humanities research on the subject of Ukiyo-e paintings. Be-
sides the presented dataset with coherent labels and annota-
tions, we also show their value in the quantification approach
to humanities research. Furthermore, we demonstrate that
machine learning models in a creative setting could address
art-style research problems.

Acknowledgement

We thank Hanjun Dai, David Ha, Yujing Tang, Neil
Houlhsby, Huachun Zhu, Justin Pinkney, and the anonymous
reviewers for their comments and helpful suggestions.

References

[Akimoto et al. 2020] Akimoto, N.; Zhu, H.; Jin, Y.; and
Conference on Computer Vision and Pattern Recognition.
[Aksoy et al. 2017] Aksoy, Y.; Aydin, T. O.; Smolić, A.; and
Pollefeys, M. 2017. Unmixing-based Soft Color Segmenta-
tion for Image Manipulation. ACM Trans. Graph.
[Art Research Center, Ritsumeikan University 2020] Art
Research Center, Ritsumeikan University. 2020. ARC
Ukiyo-e database (ARC所蔵浮世絵データベース), Informatics Research Data Repository, National Institute of
informatics. https://doi.org/10.32130/rdata.2.1.
事典). Tokyo, Japan: Tokyodo Shuppan (東京堂出版).
Aila, T. 2019. A Style-based Generator Architecture for
Generative Adversarial Networks. In IEEE/CVF Conference
on Computer Vision and Pattern Recognition.
ponica s.v. Ukiyo-e. Tokyo, Japan: Shogakukan.
[Lee et al. 2020] Lee, J.; Kim, E.; Lee, Y.; Kim, D.; Chang,
J.; and Choo, J. 2020. Reference-Based Sketch Image Col-
oration Using Augmented-Self Reference and Dense Se-
semantic Correspondence. In IEEE/CVF Conference on
Computer Vision and Pattern Recognition.
PCM. IEEE transactions on information theory.

[Pinkney and Adler 2020] Pinkney, J. N., and Adler, D.
2020. Resolution Dependant GAN Interpolation for Con-
trollable Image Synthesis Between Domains. arXiv preprint

Faces Dataset. Link.

[Renoust et al. 2019] Renoust, B.; Franca, M.; Chan, J.; Gar-
cia, N.; Le, V.; Uesaka, A.; Nakashima, Y.; Nagahara, H.;
Wang, J.; and Fujioka, Y. 2019. Historical and Modern Fea-
tures for Buddha Statue Classification. In SUMAC 2019.

[Scarselli et al. 2008] Scarselli, F.; Gori, M.; Tsoi, A. C.; Ha-

Art Colorization. Link.

[Suzuki, Takagishi, and Kitamoto 2018] Suzuki, C.; Takag-
ishi, A.; and Kitamoto, A. 2018. ‘Collection of facial ex-
pressions’ with IIIF Curation Platform - Close Reading and
Distant Reading for Style Comparative Studies. Proceedings
of IPSJ SIG Computers and the Humanities Symposium.

[Tian et al. 2020] Tian, Y.; Suzuki, C.; Claniuwat, T.; Bober-
frizar, M.; Lamb, A.; and Kitamoto, A. 2020. KaoKore:
A Pre-modern Japanese Art Facial Expression Dataset. In
International Conference on Computational Creativity.

[Tian, CODH, and ARC 2021] Tian, Y.; CODH, R.-D.; and
ARC. 2021. ARC Ukiyo-e Faces Dataset. (Created by
Yingtao Tian, ROIS-DS CODH; Collected from ARC)
DOI: https://doi.org/10.20676/00000394.
URL: https://github.com/rois-codh/arc-
ukiyoe-faces.

A.; Kautz, J.; and Catanzaro, B. 2018. High-Resolution Image
Synthesis and Semantic Manipulation with Conditional
GANs. In IEEE/CVF Conference on Computer Vision and
Pattern Recognition.
Reflective Creators

Max Kreminski and Michael Mateas
University of California, Santa Cruz
{mkremins, mmateas}@ucsc.edu

Abstract
Casual creators are a genre of autotelic, or process-focused, creativity support tools (CSTs) that prioritize the aesthetic experience of the creative process over that of the resulting product. Typically, casual creators aim to elicit a sense of ease and pleasure in their users. These are, however, not the only aesthetic experiences that a process-focused CST might reasonably aim to elicit. We introduce process aesthetics as an analytical lens through which to examine the motivations and values of different autotelic CSTs, including but not limited to casual creators. Based on this analysis, we then investigate a novel process aesthetic—reflection—for autotelic CSTs, present a set of autotelic CST design patterns intended to elicit reflection, and discuss three case studies of autotelic CSTs that make use of these design patterns.

Introduction
Casual creators are a genre of systems that “support creativity as an intrinsically pleasurable activity, rather than as an extrinsically-motivated way to accomplish tasks” (Compton and Mateas 2015). Many casual creators are based on computationally creative systems, resulting in a mixed-initiative co-creative user experience (Liapis et al. 2016). Because casual creators emphasize process over product, they are defined principally in terms of how they make their users feel during and after the creative process: exploring a creative possibility space with a casual creator should feel “fast, confident, and pleasurable”, and users should experience “feelings of pride, ownership, and creativity” when they look back on the artifacts they have made (Compton and Mateas 2015). In essence, a creativity support tool (CST) can usefully be viewed as a casual creator if it primarily aims to elicit feelings of ease and pleasure in the user.

This experience-focused definition of casual creators raises the question of whether ease and pleasure are the only, or the most important, experiential qualities that a CST could aim to elicit in its users. We contend that they are not—and, more specifically, that a CST can support autotelic creative activity without necessarily being a casual creator. The subjective experience of creation can be worthwhile even when it is not centered on feelings of ease and pleasure, and a focus exclusively on ease and pleasure might obscure or interfere with the design of autotelic CSTs that target alternative experience goals.

One alternative experiential quality that an autotelic CST might aim to elicit is that of reflection. A reflective CST might be designed to support users in carefully considering the implications of their creative decisions; introspecting on what they choose to create, how they choose to create it, and why; or attempting to refine their creative goals, intuitions, or processes through the act of creating. Users of reflective CSTs might want to create as a form of meditation or as a way of working through their thoughts and feelings on a topic, with no intention of sharing (or even preserving) the things they create. They might want to explore a design space slowly and systematically, rather than rapidly and easily converging on a specific corner of the space. And they might want to bounce ideas off of an explicitly critical computational collaborator: a trusted adversary that they can count on to push back on their decisions, or ask them to justify their actions. For these users and use cases, casual creator design patterns might not be appropriate, even as the user’s focus remains on the creative process rather than on the products that emerge.

In human-computer interaction, systems that attempt to make their users feel a specific way have been described both in terms of the “aesthetics” (Boehner, Sengers, and Warner 2008; Höök 2008) and “use qualities” (Löwgren and Stolterman 2004; Isbister and Höök 2009) they aim to elicit. In game design, the popular Mechanics-Dynamics-Aesthetics framework (Hunicke, LeBlanc, and Zubek 2004) uses the term “aesthetics” to refer to the experiential qualities of gameplay. In this paper, we will use the term process aesthetics to refer to the experiential qualities of CSTs: “aesthetics” in keeping with the terms used in adjacent fields for similar phenomena, and “process aesthetics” specifically to distinguish the aesthetics of using a CST from the aesthetics of the artifacts that a CST is used to create.

In the remainder of this paper, we first characterize reflection as a process aesthetic and argue that it is a viable aesthetic for autotelic CSTs. We then present a sampling of ten design patterns that existing CSTs use to elicit reflection. Finally, we discuss the application of these patterns to the design of three recent CSTs that embrace reflection as a design goal and conclude with suggestions for future work. We hope that our work will encourage the development of both deliberately reflective CSTs and additional new process aesthetics for CSTs beyond reflection.
Related Work

Reflection in Computational Creativity

To date, computational creativity research has primarily engaged with the concept of reflection by trying to construct computationally creative systems that are capable of reflecting on their own work. This approach to reflection dates back at least two decades, to Buchanan’s (2001) proposal of a new goal for future work in computational creativity research: the construction of computational systems that can exhibit creativity by reflecting on their own programmed limitations and finding ways to surpass these limitations through self-modification.

In a similar vein, the computationally creative system MEXICA (Pérez y Pérez and Sharles 2001) is based on the engagement-reflection model of creative writing (Sharles 1999), which treats reflection as a key part of the creative process. As a generator of “story frameworks”, MEXICA engages in a cyclic process of writing, reading what it has written, reflecting on what it has read to identify potential points of improvement, and rewriting with these points of improvement in mind.

The notion of reflection as an operation that takes place within a computational system, whereby the system reflects on its own work, also forms a key component of the Creative Systems Framework (Wiggins 2006). This framework has seen wide adoption within computational creativity research as a formalization of the field’s goals.

In this paper, however, we are not primarily concerned with building computational systems that reflect on their own work. We are instead interested in building creativity support tools that can provoke reflection in their human users, regardless of what happens within the computer. For us, a “reflective creator”—a CST intended to elicit reflection—is successful if and only if its human users find themselves drawn into reflective contemplation of their own creative goals, practices, successes, and failures. This definition mirrors the established definition of the earlier term “casual creator”: a CST that is judged successful if and only if its human users find it easy and pleasurable to use. In this sense, our view of reflection as a process aesthetic to be elicited in a human user—rather than an operation or routine that is carried out by a computational system—represents a departure from how reflection is typically viewed in computational creativity research.

Reflection as a Process Aesthetic

In The Reflective Practitioner (1983), Schon describes two forms of reflection with implications for creative practice: reflection-in-action (carried out in the moment, while a situation is still unfolding) and reflection-on-action (carried out retrospectively, once a situation has reached quiescence). Both forms of reflection are essential to the avoidance of creative impasses that result from an excessively narrow focus on one formulation of a creative problem. When a practitioner realizes that the scripts or techniques that they have been attempting to apply to a problematic situation are not yielding the desired results, it is through reflection that they can identify the essence of the mismatch, allowing them to reformulate their understanding of the problem (or of their own practices, tools, and professional role) in order to resolve the impasse. Per Compton and Mateas, casual creators are intended to expedite reflection-in-action that takes place on relatively short timescales—but reflection-on-action, and even reflection-in-action that plays out over a longer timescale, are both sidelined in the interest of keeping interactions fast and fluid. Our investigation of reflection as a process aesthetic was initially motivated by an interest in what an autotelic CST that supports reflection-on-action, or slower and more contemplative forms of reflection-in-action, might look like.

Smith (2017) surveys and critiques the design values that motivate the majority of research in procedural content generation today, while proposing reflection—as well as materiality and discomfort—as possible alternative values. In discussing the possible benefits of reflection, Smith echoes the call in HCI research for “slow technology” that deliberately prolongs interaction in order to imbue it with richer texture and deeper meaning (Hallnäs and Redström 2001).

Perhaps counterintuitively, support for reflection as an aesthetic—one that is not always compatible with ease and pleasure—can also be found in game design. Though early game design discourse often emphasized fun as the primary goal of design, recent years have seen an “opening up” of fun to reveal a broad spectrum of orthogonal or even contradictory possible aesthetics. Just as personal and queer games often deliberately reject fun in favor of alternative aesthetics, including discomfort and frustration (Anthropy 2012), autotelic CSTs could equally embrace process aesthetics beyond ease and pleasure. One could imagine, for example, a critical CST that highlights how design decisions impact human participants in the supply chain, just as Molleindustria’s The McDonald’s Videogame seeks to emphasize the human impact of the business practices it models.

Autotelic Creativity Support Tools

Nakakoji (2006) divides creativity support tools into three categories: those analogous to running shoes, dumbbells, and skis. Running shoes aim to provide additional support for a well-understood activity (running) with obvious criteria for success; dumbbells are used to develop creative capacity in the user without being employed to produce creative artifacts directly; and skis attempt to enable a new form of creative activity (skiing) that would not be possible without the tool. Because skis are intended to enable new activities, they can be hard to evaluate initially if other tools enabling the same experience don’t already exist. Process aesthetics present one possible strategy for evaluating tools of this nature in terms of whether they successfully enable an experience with particular subjective qualities.

Dumbbells hint at another possible justification for reflection as a process aesthetic. Krakauer (2016) extends Norman’s description of computer systems as cognitive artifacts (Norman 1991) by drawing a distinction between complementary cognitive artifacts, which build up capabilities in the artifact’s human users that remain even when the ar-

http://www.molleindustria.org/mcdonalds/
tifact itself is removed, and *competitive* cognitive artifacts, which replace or displace the user’s capabilities. From this perspective, reflective creators that can assist users in developing a tool-independent reflective creative practice could be viewed as complementary cognitive artifacts. However, this is not the only reason that reflective creators might be valuable. Like ease and pleasure, reflection can be an instrumental value for CST designers—judged as useful because it leads to the production of more or better artifacts or creators—but it can also be a terminal value, judged as inherently worthwhile.

In her dissertation, Compton (2019) discusses the difficulties associated with “slow creators”, which have a wide gulf of evaluation due to inherent limitations in the speed of the underlying computationally creative system. However, this analysis does not consider the possible value of slowing down the creative process in order to promote reflection.

Petrovskaya, Deterding, and Colton (2020) survey existing commercially available casual creators and categorize them according to their main interaction technique. The findings of this survey seem to support the claim that most commercially available apps fitting the definition of casual creators are optimizing primarily for ease and pleasure, and especially for *speed* of creation in support of this goal.

Nelson et al. (2018) suggest that at least some users of casual creators are motivated primarily by curiosity, either about the tools themselves or about the generative spaces that these tools allow their users to access. The user behavior patterns documented in this work may imply that some users of casual creators actively seek out a reflective creative experience, rather than an easy or pleasurable one.

### Design Patterns

How, concretely, can CSTs be designed to elicit reflection? In this paper, we identify an initial set of ten promising design patterns drawn from existing systems. Design patterns, as introduced by Alexander (1977), are high-level descriptions of solutions to problems that frequently recur in a particular design space; here, we present patterns that apply to the design space of CSTs and the recurring problem of eliciting reflection in the human users of these tools.

Like the accounting of casual creator design patterns provided by the original casual creators paper (Compton and Mateas 2015), this list of reflective creator design patterns is not intended to be exhaustive. Instead, we aim to represent features that are commonly found in existing reflection-focused CSTs, and that we have found useful in our own design analysis of CSTs (especially the three case studies discussed in the following section).

**Reifying intent** Many reflective creators ask their users to make their creative intent explicit and provide a set of mechanisms for describing and negotiating intent. Often—but not always—these tools provide an *intent language* (Martens and Hammer 2017) that allows users to describe their intent in a systematic, machine-parseable fashion.

Asking users to make their intent explicit can promote reflection even when the intent is not understood by the machine. In a multi-user context, reifying the design intent makes it a shared object of comment between the users and enables metacommunications about what an artifact should and should not include. And even in a single-user context, the mental work of identifying and expressing one’s intent requires reflection on one’s own goals, values, and priorities, regardless of whether the intent is then fed into a computational system.

One multi-user example of this pattern can be found in the tabletop storytelling game Microscope², which provides a feature called the *palette* that affords negotiation among players as to what they would and would not like to see happen in the story. Similarly, PolicyKit (Zhang, Hugh, and Bernstein 2020) provides internet communities with an intent language for describing and negotiating moderation policies, prompting members of these communities to reflect on and openly discuss how they want to be moderated and why. Here, however, the intent language is fully system-understandable, enabling the underlying “policy engine” to automatically enforce agreed-upon policies.

**Elaborating intent** Once the user has specified their intent in a systematic way, an obvious next step is to generate and display many possible realizations of that intent. In so doing, a computational tool can attempt to present the user with information on potentially unexpected ramifications of the intent; contradictions hidden in the intent; or discrepancies between the envisioned and actual consequences of the intent as directly specified.

This is central to the approach taken by the game design support tool Germinate (Kreminski et al. 2020c), which uses generative methods to translate a user’s initial high-level rhetorical intent into a variety of playable digital games. A similar approach is taken by creative writing support tools Writing Buddy (Samuel, Mateas, and Wardrip-Fruin 2016) and *Why Are We Like This?* (Kreminski et al. 2020b); these tools both allow users to specify and modify their storytelling goals, then suggest possible character actions that might help to advance these goals.

**Inferring intent** Based on the creative decisions a user has made, the system can also attempt to *infer* their intent and display it to them—sometimes as a set of sentences in a formal intent language, sometimes merely as a list of adjective labels. The difficulty of translating an implicit creative intent (which the user does not yet fully understand) into a explicit intent can thereby be mitigated: it is usually easier to accept or reject specific system-suggested assertions about an intent than it is to write out an explicit statement of a formerly purely implicit intent from scratch.

Germinate infers intent from mixed user-specified and system-generated game rules via *proceduralist readings*, enabling it to suggest new high-level design goals to the user based on decisions they have already made.

Mahajan et al. (2019) offer automated critiques of student-created branching narrative projects by comparing the input project to a database of existing projects on a variety of descriptive criteria. Their system then produces a report for the student on which existing projects were most

---

²http://www.lamemage.com/microscope/
and least similar to theirs and on what criteria (including frequency of choices, density of text, and overall length) their project stands out relative to others. This report can be read as an inference of intent, allowing the student to understand how their project might come across to the reader and either lean into or modify their approach based on whether the inferred intent matches their preferred direction.

One promising technical approach to inferring intent involves the use of discriminative learning, as discussed by Karth and Smith (2019) and Kreminski, Wardrip-Fruin, and Mateas (2020). Based on a set of user-provided example artifacts, a co-creative system can infer intent from the shared characteristics of these examples, generate more examples based on the inferred intent, present the generated artifacts to the user, and let the user approve or reject the generated examples to progressively refine their intent.

**Interpretive refraction** One way to facilitate reflection is via defamiliarization, or deliberate creation of distance between creators and their practices or artifacts when they would otherwise be “too close” to see the flaws. To achieve this defamiliarization, it can be helpful to display to the user multiple different computational judgments or readings of the work-in-progress artifact, even when these readings seem to point in mutually incompatible directions.

In contrast to the entertaining evaluations often employed by casual creators, reflective creators more frequently employ evaluation methods that are intended to be taken seriously. Additionally, in order to ensure that real co-interpretation (Pousman et al. 2008) takes place between the user and the system, the system’s readings should involve some actual analysis of the creative artifact or process; they should not be wholly disconnected from the user’s input, as is sometimes the case in casual creators (such as BECOME A GREAT ARTIST IN JUST 10 SECONDS⁵, which presents a humorous but baseless “similarity score” between the user’s glitch-art creation and a famous classical painting.)

Smith et al. (2015) leverage interpretive refraction to defamiliarize the process of textile crafting. Their crafting tool sonifies the user’s physical gestures to provide a new, unfamiliar feedback channel and thereby prompt reflection on practice. Also in the domain of physical fabrication, Fabricaide (Sethapakdi et al. 2021) prompts users to consider the material costs of their design by calculating how much of what materials would be needed to create it and visualizing this information in real time.

Sentient Sketchbook (Liapis, Yannakakis, and Togelius 2013) uses visualizations of different “fitness dimensions” to highlight different aspects of game design. This idea has been generalized somewhat in the context of game design support under the label of “computational critics” (Osborn, Grow, and Mateas 2013): systems that examine different aspects of a game design and offer criticisms.

**Contextualizing choices** In Schön’s account of creative design, the design process involves “spinning out a web of [...] implications” (Schön 1983) in which each individual design decision may impose far-reaching constraints on other aspects of the design. Because these constraints have potentially nonlocal impact, and because designs are often so large that it is impossible to attend to all of a design’s features and implications at once, it can be easy for design moves to quietly invalidate other choices that have been made during the design process, often in a way that is not immediately visible to the designer. Therefore, it may be especially important for reflection-focused CSTs to support their users by highlighting the implications of decisions.

For instance, if the computer is capable of understanding the impact of a single choice in terms of multiple different aspects of the high-level design intent, it can inform the user when a choice that appears to be a good one for local reasons (perhaps because it advances one particular design goal) also has negative impacts on the realization of other design goals, which the user might not currently be considering. If the design space is understood as containing choice points, as Schön suggests, then decisions that cut off certain parts of the design space might also be useful to highlight. This kind of feedback might be especially easy to provide when the underlying generative system already understands the creative process in terms of the navigation of a design space, as (for instance) in design space modeling approaches to procedural content generation (Smith and Mateas 2011). Finally, a system that understands the impact of individual moves might also use that information to show creative decisions in context, among other strong possibilities (for instance by prompting the user with alternative moves that they could reasonably have chosen to perform instead).

One example of this approach can be seen in the work of Kybartas, Verbrugge, and Lessard (2020): a co-creative narrative system that operationalizes the possible worlds theory of narrative structure to identify a “tension space” consisting of conflicts between characters’ ideal world states and the state of the world as it actually exists. This tension space is then visualized as it evolves throughout the process of authoring an emergent narrative storyworld, making it apparent to the user when a creative decision that they have made substantially increases or reduces narrative tension.

**Challenging choices** One of the editor’s roles in the creative writing process is to constructively push back against the writer’s decisions. Similarly, some users deliberately seek out or implement CSTs that help them catch and eliminate their bad habits, for instance by automatically detecting and flagging uses of specific words or sentence structures.⁶ Computationally creative systems with a deeper understanding of how creative artifacts are structured could extend this kind of reflection support to other domains.

Many co-creative systems already base their actions (or action suggestions to the user) on a model of what creative decision would be most likely at this point in the creative process. For instance, language model-based CSTs for writing often use likelihood to recommend words or phrases to the user (Manjavacas et al. 2017), while some game design tools do the same for level design decisions (Guzdial et al. 2019). This same information could be used to push

---

⁵https://igf.com/become-great-artist-just-10-seconds

⁶http://matt.might.net/articles/shell-scripts-for-passive-voice-weasel-words-duplicates/
users away from making clichéd decisions, for instance by informing them when they perform a highly likely action and suggesting less-likely alternatives.

Like the previous design pattern (contextualizing choices), this pattern can increase users’ doubt in their own choices and slow their exploration of the design space considerably. As a result, both patterns might be considered undesirable in many casual creator contexts. In a reflective context, however, introducing doubt and slowdown may be exactly what is needed.

Reflective encoding Some CSTs that support the construction of generative models are sometimes used not to build realistic or directly useful models, but instead to assist users in reflecting on the phenomenon they are modeling. One example is the social simulation tool Ensemble (Samuel et al. 2015): on at least two occasions, it has been used to model real-world social phenomena with the goal of reflectively developing a better understanding of that phenomenon, but without regard for the direct applicability of the resulting model (Dickinson, Wardrip-Fruin, and Mateas 2017). Critically, even when these models are not immediately applicable to any existing problem, the act of constructing them—of formalizing knowledge sufficiently that it can be encoded in a relevant notation—prompts reflection and a deepening of understanding within the model’s creator. This is reminiscent of the autotelic uses of formal languages (such as baseball scorecards) discussed by Nardi (1993): some baseball fans find that their understanding and enjoyment of the game is deepened when they follow along by reflectively encoding the action of the players into a formal language.

Though the generative text tool Tracery (Compton, Kybartas, and Mateas 2015) is often cited as an example of a casual creator, some of its more advanced features (such as actions, which allow generated substrings to be saved and reused) sometimes compromise the aesthetic of ease. However, these features are key to enabling the form of reflective encoding for which Tracery is often used: reflectively building up a Twitter bot or other text generator to imitate a particular corpus, essentially conducting a manual generativist reading of the corpus in question (Kreminski, Karth, and Wardrip-Fruin 2019).

Future encoding-focused reflective creators might prompt the user to incrementally flesh out the model they’re building, perhaps by identifying underspecified parts of the model and asking pointed questions about them (as in Garbe’s proposed worldbuilding assistant chatbot).3

Reflective enactment Some computationally creative systems, especially in the domain of textile crafts, generate designs that can only be physicalized by human labor. Though this could be seen as a weakness of these systems, Albaugh et al. (2020) suggest that the underdetermination of computer-generated designs can also be viewed as a resource for promoting creative reflection. Laborious enactment of computer-generated instructions can prompt reflection on the meaning and significance of the crafting process and the crafted artifact alike.

Embroidered Ephemera (Sullivan 2020) is a computationally creative system that generates an embroidery sampler design from a user-selected tweet, but leaves the work of actually embroidering a generated design to the user. Though the system was initially conceived of as a casual creator, its author reports that the time-consuming nature of embroidery work compromises the aesthetic of casual creation in some regards. We argue that this is because Embroidered Ephemera instead exemplifies an aesthetic of reflection in this aspect of its design. By contrasting the low time cost of selecting a tweet to feed into the system with the high time cost of embroidery, the system creates a moment of reflective commitment at the time of tweet selection, encouraging the user to carefully consider why they might want to physicalize this particular tweet in this particular way at this particular time.

Reflective repair Reflective enactment leverages the incompleteness of computer-generated designs to prompt reflection through the process of completing them. Reflective repair deepens this focus on incompleteness by introducing incorrectness to the computer-generated designs as well, requiring users of a computer-generated artifact to fix up minor problems or fill gaps left by the computer. Repair is an essential component of Sharples’s model of reflection in creative writing (Sharples 1999), and encouraging users to engage in thoughtful repair of flawed artifacts may be an especially useful strategy for promoting the development of reflective capacity that persists in users beyond their experiences with a particular tool.

The SkyKnit system (Shane 2018), which uses machine learning to generate knitting instructions, provides a key example of reflective repair in action. Because the instructions that the system generates are often flawed, knitters who attempt to realize these designs are frequently forced to improvise repairs to nonsensical aspects of otherwise-acceptable plans. This can result in a productive kind of defamiliarization, challenging knitters to think outside the box and sometimes even invent new stitch types in their attempts to repair machine-generated designs.

Reflective revisitation Frequently during the design process, a designer’s attention is fixed on one specific, narrow part of a larger, more complicated design situation. This can introduce problems when design decisions made in one part of the situation subtly invalidate decisions made elsewhere, but the designer has not yet noticed the conflict. To combat the tendency for earlier decisions to linger even when they no longer fit the design situation as a whole, a computational creative partner might explicitly prompt its user to re-engage with or reevaluate decisions they made in the past.

This might take several forms. Building on the pattern of challenging choices, the CST might actively seek out decisions that were made some time ago and prompt the user to reconsider the decision in light of how the design as a whole (including the design intent) has evolved since then. Additionally, a CST might gradually introduce and tighten artificially imposed constraints on the design in order to force periodic reevaluation of decisions; similar patterns are em-
Redactionist went through three distinct stages of development. It initially took the form of a push-button generator called Blackout (Kreminski, Karth, and Wardrip-Fruin 2019) that, given some input text, would produce a poem by erasing words from the input without further user interaction. To make the system a better casual creator, it was then updated with a mixed-initiative interaction model. Given a block of input text, the system would scan the text from left to right and come up with three possible next words for the poem to include. Then it would present these options to the user, wait for a selection, and continue scanning left-to-right, repeatedly calculating three more options based on the user’s previous choices until the poem was complete. Limiting the order of text traversal from left to right and restricting the user to three choices at each step was intended to limit overwhelm, in keeping with the prioritization of ease-of-use over fine-grained control in casual creator design.

However, this version of the system was found to be unsatisfying. Despite the potential for greater overwhelm, removing some of the artificial restrictions on the possibility space of each poem turned out to produce better results, especially when combined with a deliberate application of the contextualizing choices design pattern. In the most recent version of Redactionist, the system identifies up front a set of words that are valid for inclusion in the poem and presents the user with the initial choice to select any one of these words. The selection of a word constrains what patterns might be viable matches for the selected set of words, restricting future choices somewhat. At any time, any selected word may be unselected again, possibly removing constraints and enabling some other words to be selected instead. This turned out to produce a compelling reflection-focused creative experience with an ideal balance of constraint and freedom: though the user now faces many more possible options at the beginning of the creative process, they can easily see which choices are cut off when they select a specific word for inclusion, and this prompts careful deliberation over which words are most essential to the intended meaning of the poem.

How could this experience of reflection be improved even further in the future? At present, the computational system in Redactionist is solely responsible for determining which words from an input text are selectable for inclusion in a poem, using a (somewhat flawed) part-of-speech tagging process. Adding a reflective repair step at the end of the poem creation process wherein all of the words in the input text become available for inclusion might help to preserve the initially helpful computational mediation of the creative process (which helps the user develop a stronger sense of what kind of poem they would like to create) while also allowing the user to deviate somewhat from the computer’s idea of what constitutes a valid poem once their creative intent has been clarified.

Germinate
Germinate (Kreminski et al. 2020c) is a mixed-initiative CST for digital games that make arguments through procedural rhetoric (Bogost 2010). It presents the user with an interface for specifying the high-level rhetorical argument that they want to make through gameplay; a means of automatically transforming this argument into a variety of specific, playable digital games; and affordances for modifying the
Germinate already implements the three intent-related patterns \textit{(elaborating, reifying and inferring intent)} described in this paper. It also implements \textit{interpretive refraction} via the proceduralist readings of generated game rules that it conducts and surfaces to the user as a high-level summary of game dynamics. Additionally, Germinate facilitates \textit{reflective repair} by allowing users to select specific rules and mechanics from flawed generated games and extract them directly into the design intent, enabling the preservation of these rules and mechanics in future generated games even as the rest of the design evolves.

However, Germinate’s ability to facilitate reflection on how high-level design intents can be expressed via low-level game mechanics is limited by its current design. In particular, since the rules governing the system’s understanding of how mechanics work together to create aesthetics are fixed and opaque to the user, the system may repeatedly attempt to realize a user’s high-level intent through combinations of mechanics that do not actually support the intended player experience from the user’s perspective. The system could more effectively support reflection on intent by opening these rules up to \textit{reflective encoding}: perhaps first allowing the user to view the system’s reasoning as to why it interprets a particular combination of mechanics as creating a particular target aesthetic, then letting the user disable interpretive rules that they disagree with or even introduce new ones as they develop their intent.

To support this process, an updated version of Germinate could implement \textit{challenging choices} and \textit{reflective revisitation} by occasionally prompting the user to annotate specific creative decisions with which aesthetic goals these decisions support. This might occasionally provoke users to realize that some of their choices do not effectively support the aesthetic goals that they are currently pursuing, maybe prompting a revision of the interpretive rules.

Additionally, as further support for \textit{inferring intent}, a future version of Germinate could make use of discriminative learning for intent refinement. As in (Karthis and Smith 2019) and (Kreminski, Wardrip-Fruin, and Mateas 2020), users could be asked to accept or reject generated games based on their alignment with the high-level design intent, and refinements of intent could be inferred from the shared characteristics of the accepted and rejected games.

\textbf{Why Are We Like This?}

\textit{Why Are We Like This? (WAWLT)} (Kreminski et al. 2020b) is a playful, multi-user mixed-initiative CST for creative writing, powered by a social simulation engine that governs the behavior of a small cast of simulated characters. The system suggests actions for characters to perform, based on a model of character motivations and player-provided storytelling goals, and players choose which of these actions they would like to realize. Terse, system-generated descriptions of these character actions are then added to a running transcript of the story so far, which can be further annotated by the players with a more detailed description of each action.

In its current form, \textit{WAWLT} provides support for \textit{reifying intent} (by allowing users to specify explicit “author goals” for what they would like to happen next in the story) and \textit{elaborating intent} (by suggesting actions that might fit the currently selected author goals). \textit{WAWLT} also attempts to implement a limited form of \textit{contextualizing choices}, both by showing each system-suggested action among several reasonable alternatives and by highlighting which author goals these suggested actions would immediately advance. This latter feature could also be viewed as a form of \textit{interpretive refraction} in which the system evaluates prospective actions from the perspective of multiple distinct author goals at once.

A small-scale user evaluation of \textit{WAWLT} (Kreminski et al. 2020a) found that some users struggle with a lack of clear direction when using the system, and sometimes forget to update their design goals as they move through the creative process. Taken together, these findings suggest that the system could support the progressive \textit{refinement} of creative intent more effectively than it currently does. To that end, from a reflective creators perspective, we believe that \textit{WAWLT} would benefit from a reworking of its interaction model to tie the reflective revision of intent more deeply into its core interaction loop.

By \textit{inferring intent} on the basis of events that the user has already selected, a future version of \textit{WAWLT} could more proactively identify and surface the goals that the users appear to be pursuing, and thereby prompt them to update their stated intent as their goals change. Additionally, an implementation of the \textit{challenging choices} design pattern could help to provide users with a stronger sense of direction by making it clear to users when a proposed action \textit{inhibits} one or more active author goals, or perhaps when the users have tunnel-visionsed on the advancement of one author goal for several turns at the expense of others.

From a technical perspective, the implementation of \textit{WAWLT}’s underlying generative system as a non-reversible simulation inhibits the ability to develop features that pro-
mote reflective revisitation and reflective repair. Past actions cannot be modified lest they implicitly invalidate future actions, so the system cannot prompt users to change the decisions contributing to their story’s notional past. However, because the system allows users to write free text annotations for each event that has transpired, and because these annotations are not reasoned over by the system in any way, this may serve as an effective escape hatch for revisitation and repair: users can be prompted by the system to rewrite their descriptions of past events (perhaps to include some foreshadowing) if these events turn out to be pivotal later on in the story, or if they contribute to a high-level trend in the storyworld that is later reversed (such as two historically hostile characters eventually becoming friends, perhaps implying that the severity of past hostile interactions between these characters should be downplayed).

**Conclusion**

Not many reflective creators yet exist. However, it is our hope that by giving this category of systems a name and drawing together some design patterns demonstrated in existing examples, we will begin a conversation that leads to the development of more reflective creators going forward. The recent success of the “casual creators” label in drawing together practitioners and researchers with an interest in autotelic creativity support tools is inspiring to us in this regard, especially in light of the variety of work presented at the first Casual Creators Workshop at ICCC last year.

Many of the design patterns we have discussed in this paper add friction to the creative process. In our view, this is not necessarily a bad thing. Reflection is a viable process aesthetic in its own right, distinct from ease and pleasure; though approachability-focused CSTs may still want to prioritize ease and pleasure to eliminate barriers to entry, some users will always want to engage in reflection for the sake of reflection, even when it is challenging.

The development of reflective creators will not necessarily result in better creative works, especially short term. However, it might result in more thoughtful creative practices and practitioners.

Beyond ease, pleasure, and reflection, we believe that many other viable process aesthetics for autotelic creativity support tools remain to be discovered. By identifying one novel process aesthetic for CSTs, we hope to encourage other researchers to seek out further new aesthetics that are of interest to them.

**Acknowledgements**

Thanks to the Seabright Camerata, and especially Melanie Dickinson, for feedback on early drafts of this paper. Thanks also to the anonymous reviewers for their feedback, which led to substantial clarification of our argument.

**References**


Metaphor, Blending and Irony in Action:
Creative Performance as Interpretation and Emotionally-Grounded Choice

Tony Veale
School of Computer Science
University College Dublin
Dublin, Ireland.
Tony.Veale@UCD.ie

Philipp Wicke
School of Computer Science
University College Dublin
Dublin, Ireland.
Philipp.Wicke@UCDConnect.ie

Abstract
Metaphor is a powerful tool in the performer's tool box, not least because it can operate at several levels at once. As our linguistic metaphors deliver rhetorical flourishes, conceptual metaphors change the way we see the world, while our metaphorical body language allows us to take postures and stances that are both figurative and literal. Each kind of metaphor is another expressive choice that creative performers can make to convey their meanings. This choice sits at the heart of what it means to be a creative agent, for agents who lack choice and do exactly as they are told cannot exercise genuine intent for their actions. We shall explore how interpretation is wrapped up with choice to appreciate and to achieve emotional creativity in a system for generating and enacting automated tales. We explain here how three levels of description – linguistic, conceptual and physical – are integrated in a framework that motivates the use of metaphor or irony as a creative choice, by robot performers aiming to go beyond a predetermined script.

Interpretation Makes It So
Hamlet reminds us that “there is nothing either good or bad but thinking makes it so.” We form our judgments, creative or otherwise, by viewing situations and events through the lens of interpretation. It is a capacity for interpretation that allows actors to deliver a creative performance even when their actions and their lines are dictated for them. Indeed, it is interpretation that allows scripted performers of any kind to do more than obediently follow the script they are given, and to bring something of themselves to their work. It is in the gap between interpretation and execution that creativity can take root and blossom, even for the most scripted roles. When performers interpret a script to create a performance, or otherwise, by viewing situations and events through the lens of interpretation. It is a capacity for interpretation that we set out to explore here: metaphorical exaggeration, or even an ironic response, not as to avoid repetition and foster variety, it is making a hollow choice based on form without meaning. This is the essence of what is often called mere generation (Veale 2012; Ventura 2016): the glib production of well-formed outputs that are valid only because they obey the rules of the game, not because they have an inherent value that is appreciated by the generator itself. Mere generation uses generic rules to make specific choices, but leaves the interpretation of the specific ways in which the rules are instantiated to the user.

The dichotomy between freedom of choice and no choice at all is an extreme one, but there are more subtle dilemmas. There is, for one, a real distinction between true choice and empty choice, or choice for its own sake. At any given time an agent may have several valid responses available to it, thus offering it an opening for a creative choice. If an agent chooses randomly from these possibilities, or if it chooses so as to avoid repetition and foster variety, it is making a hollow choice based on form without meaning. This is the essence of what is often called mere generation (Veale 2012; Ventura 2016): the glib production of well-formed outputs that are valid only because they obey the rules of the game, not because they have an inherent value that is appreciated by the generator itself. Mere generation uses generic rules to make specific choices, but leaves the interpretation of the specific ways in which the rules are instantiated to the user.

The rules of a merely generative game are themselves a script into which formal choices like these can be baked in from the outset. A disjunctive script that says do this or this or that can be just as rigid as one with no disjunction at all. Choice without interpretation leads to empty variations that thwart self-evaluation and lack creative force. We critique such an approach to story-based performance here, one that uses disjunction to achieve diversity of output without truly appreciating the meaning of that diversity. With this as our baseline, we model what it means for a performer's choices to be driven by an emotional understanding of the script. A performer should choose to react to a narrative event with a metaphorical exaggeration, or even an ironic response, not just because this is a possibility, but because it enhances the telling of the story to react this way. This will require us to insert an emotional layer between the conceptual layer of the system – its plots, actions and model of cause and effect – and its expressive layer of physical gestures and spoken words. This sandwich of distinct layers will allow performers to make informed choices that are grounded in context.

Interpretation is the missing ingredient that fills the gap between mere generation and intentional creation. It is key
to a producer’s efforts to make informed choices, just as it is key to a consumer’s efforts to ascribe value to any outputs. Interpretation recognizes the value of a departure from the scripted norm, but interpretation is more than recognition. To appreciate the difference between these related notions, consider the following example of an email communication from the first author to the second, as sent via Gmail:

We went to see “Tenet” last night. Our brains are still bent out of shape. But we plan to see it again last week, so we’ll understand it eventually.

The joke, such as it is, relies on the specifics of “Tenet”, a time-travel movie in which mysterious agents use reverse entropy to go back in time. The statistical language model used by Gmail (Chen, Lee, and Bansal 2019) to predict future texts and flag possible errors in past texts spies a rather improbable word choice in “again last week,” and it dutifully underlines “last” in blue to signal its recognition of this departure from the norm. But the model does not “get” the joke – it sees no link between “Tenet” and the future-looking use of “last week”– and so cannot interpret it as irony. This is a consumer view of a producer’s attempt at wit, but what of the producer’s perspective? Let’s choose a more normative situation that relies only on general world knowledge:

I have never married, but I have had a few near misses.

To re-package this metaphor as a joke, we might replace the high-probability word choice of “misses” with the phonetically identical, but much more improbable, “Mrs”:

I have never married, but I have had a few near Mrs.

The replacement yields a recognizable departure from the norm that a language model like Gmail’s can easily detect, but mere recognition of this departure is not enough for wit. We could, for instance, have replaced “misses” with “kisses” or “fishes” to achieve the same low-probability punning surprise, but such a replacement would make little or no sense. It takes interpretation to appreciate the logic of “Mrs” in this context, since only “Mrs” produces the same humorous kick. Ideally, the interpretation of the producer will mirror that of the consumer, to predict the surprise and its final resolution.

The rest of the paper explores how an interpretative layer can be inserted into an existing computational framework for the embodied performance of machine-generated stories. We begin by considering this existing system as a baseline, to ascertain the degree to which its choices are disjunctive but empty, or interpretation-driven and potentially creative. We aim to shift its workings closer to the latter by basing its decisions on an emotional understanding of a story’s events. Disjunctive choice can be a useful source of plot variation if the disjunction is motivated by the emotions of the story’s characters or an audience’s attitudes to them. We show how a balance is realized by allowing audiences to influence the plot as they reveal their own feelings for a story’s characters. But what matters is a performer’s choices and how they are interpreted by the audience. As embodied actors that bring a tale to life with their gestures and spatial movements, these choices must be seen as coherent for the plot. We present a crowd-sourced evaluation that shows this is indeed the case.

A Critique of Pure Disjunction

The Scéalextric framework of (Veale 2017) adopts a story grammar approach to plot generation. Its tales are assembled in a click-&-build fashion from a large set of pre fabricated plot segments of three successive actions apiece. Inspired by the Plotto framework of (Cook 1928) – one of the oldest structuralist approaches to systematic story construction – a Scéalextric plot is first assembled by linking a series of plot triples end-to-end, or by recursively refining a single over-arching triple in a top-down fashion to flesh out a narrative arc. Cook listed almost 1,600 plot triples in his 1928 book, while Scéalextric’s stock numbers over 3,000, each crafted from a set of 800 verbs that relate the generic characters A and B. As this genesis in Cook’s approach will testify, Scéalextric does not employ a particularly novel approach to plotting. Rather, its appeal lies in its scale, modularity and openness. A skeletal plot is easily rendered in English with a large set of idiomatic renderings that map actions from the logical to the text level, via forms that include idioms and metaphors. The system’s data-rich modules are open for all to use, and new modules – to e.g. add dialogue, as in (Wicke and Veale 2020), or to append a moral – are easily defined.

In top-down mode, the initial triple is used to give shape to a story. In end-to-end mode, a new triple is added to the growing plot if it shares a connecting action, and if no loop or repetition results from its addition. So, the choices made during plot assembly are formal ones, decided on the basis of compatibility with the story grammar rather than for any semantic or narratological reasons. These wholly structural decisions require no interpretation of the evolving plot, and so can be considered merely generative disjunctive choices. As such, Scéalextric exhibits broad generativity but little or no true creativity in the assembly of its plots or in the final rendering of those structures as idiomatic English stories.

So where does the creativity, if any, arise in this system? As described in (Veale 2017), Scéalextric goes beyond the purely disjunctive to choose the characters that will fill the A & B positions in its plot skeletons. A database of familiar characters from fiction and history, called the NOC List, is used to fill each role with a recognizable personality, so that each character’s extensive backstory (as stored in the NOC) can be woven into the rendering of the tale. For instance, if the plot calls for A to insult B, the idiomatic rendering of insult offers a generic account of the offense, but the NOC allows specific negative details of the target to be aired too. The NOC also allows metaphors and similes to be coined on the fly, so that e.g. Richard Nixon insults Bill Clinton by likening him to Pepé Le Pew, or insults Frank Underwood by comparing him to Keyser Söze, thus winkingly breaking the fourth wall at the same time (as the NOC knows that each character was portrayed on screen by the same actor).

Crucially, NOC characters are chosen for their suitability to specific actions in the plot. If the plot turns on a betrayal, a sneaky character is chosen; and if one character must heal another, a doctor is chosen. To achieve a measure of wit, A & B are instantiated as a pair, and this is where Scéalextric makes semantic choices that are guided by interpretation. As outlined in (Veale 2017), character pairings are chosen to exhibit a mix of appropriateness and incongruity, or what
humour theorist Elliott Oring calls appropriate incongruity (Oring 2011). As in the last/next and misses/Mrs oppositions explored in the introduction, a good pairing produces a reassuring surprise, an apparent mistake that only makes sense on closer examination. Two NOC entities are paired if they are linked in the popular imagination — perhaps they were portrayed by the same actor, or created by the same author, or belong to the same group, or share some key properties — and there is also some common-sense bar to their union, such as that one is historical and the other fictional, or they belong to different fictional franchises or historical periods. *Scéalextric* might pit Alan Turing against Sherlock Holmes, or make business partners of Ada Lovelace and Steve Jobs. The effects of each high-friction pairing percolate through the rendering of the tale as a whole, reminding audiences of the appropriate incongruity at the story’s core, as aspects of each character — properties, clothing, or physical settings — are integrated into the surface renderings of plot actions.

*Scéalextric* has been used as a generative basis for other story-telling systems. Wicke and Veale (2018) used robots to enact its tales with physical gestures and spoken dialogue. Veale, Wicke and Mildner (2019) subsequently built a model of performance, called *Scéalability*, around this generative core, allowing for tales to be physically enacted by a cast of robots and smart devices. Anthropomorphic robots move about a stage as they act out the central roles of a story, while an omniscient narration is voiced by an Amazon Echo/Alexa. Additional dialogue is layered over the textual rendering of each story, so that the robots have apt lines to speak as they move and gesticulate. The dialogue module that generates this spoken script is powered by simple disjunctive choice: for each of the 800 actions in the *Scéalextric* plot vocabulary a set of dialogue fragments for A and for B is defined, and the actors choose randomly from this set for a given action. This simple approach is especially effective in stories with more than two characters. Since roles such as A-spouse and B-friend cannot be physically enacted with just two robots, we know of their actions only from the narrator, and from the commentary of the main actors as they react to events.

The physical actions of the *Scéalability* robots are also determined by simple disjunctive choice. For each of the 800 possible actions in a *Scéalextric* plot, a set of motor scripts is associated with the A and B roles, and the robot performers are free to choose which script to execute. As in the dialogue, these choices are always constrained by the needs of the current action. No consideration of past actions, and of how they influence an audience’s interpretation of the current action, is brought to bear, and no freedom is given to depart from the script. This prevents the performers from interpreting their scripts, to decide that a certain plot point needs to be emotionally heightened with metaphor or irony. We aim here to remedy this lack of interpretative freedom.

**Plot Disjunction at Time of Performance**

A story grammar is essentially a causal graph of actions and their consequences. A “walk” through this graph, whether a random walk or a goal-directed journey, yields a single path and a single plot line. Branching points in the causal graph present choices that are resolved at the time of the walk, not at the time of the story’s performance for an audience. But this need not be the case: if branch-points are inserted into the plot, turning it from a line into a tree, those choices can be resolved latter, perhaps with the help of the audience.

When a story generator makes these choices for itself, by treating each branch point as a purely disjunctive choice, it simply explores the space of possible stories without regard for the emotions of its characters and those of an audience. Interpretation is supposed to offer insights on such matters to the performers, but by this time the tale has been written. By allowing choices to be made at the time of performance, an interpretation of what has gone before in the narrative can shape the course that the performers will take. Indeed, the performers can involve the audience in their decisions, so that they make choices that seem emotionally plausible.

We can use the word “script” to denote the sequence of actions to be followed by a performer, or a body of code to be executed on a machine. As we have seen, each involves a different idea of “interpretation.” While the latter brooks no flexibility, no metaphor, and no loose readings of the text, it does allow for conditional *if*-then-*else* branching structures. To support performance-time decisions regarding plots, we incorporate both senses of “script” into *Scéalextric* stories. It is a simple matter for a story-grammar to generate *if*, *then* and *else* markers in its plot lines, and to recursively expand different plot continuations after a conditional branch point. The resulting plot is still a linear sequence of symbols, but, like a computer program, it is executed by its performers in a dynamic, non-linear fashion. When robot actors resolve a branch point for themselves, they can use the interpretative, emotion-based mechanism we present in the next section.

Or they can ask an audience to provide an interpretation for them, falling back on their own logic when none is offered. Certain plot actions represent dramatic choice points in a story, as when, for instance, A considers forgiving B for an earlier offense. It is at these points that the story grammar obtains maximal benefit from a disjunctive turn in the plot, as these emotion-laden turning points should also elicit an emotional response from the audience. To elicit a response, the performers explicitly ask the audience for their input at these junctures, by e.g., asking “Should I forgive this guy or not?” To register this response, if one is forthcoming, a video camera is used to capture the facial expressions and the hand gestures of audience members. Since the robots themselves use gestures to convey their emotions, it seems fitting that the audience likewise joins in the performance.

The robot whose character is to perform (or not) the given action pauses, turns to the audience, and poses its question. The system’s camera is constantly trained on the audience, but its video feed is only examined in the moments after the question is posed. The robot also gestures to signify that it awaits an answer, but is capable of carrying on without one. Two distinct neural networks examine the same images: the first (Cao et al. 2018) scans the video for hand gestures, returning a label, a bounding box and a confidence score; the second (Goodfellow et al. 2013) scans the image for facial emotions, likewise returning a label, a location and a score (see Fig. 1). The performer makes its decision on the basis of both data sources as weighted by their confidence scores.
Once More, With Feeling!

Tapping into an audience’s reactions in this way allows the actors to borrow their emotional interpretation and make it their own. However, when those reactions are not apparent, the performers must arrive at their own interpretation of the current state and the actions that led to it. In fact, performers must do this for themselves anyway, for all of the other states that do not correspond to an explicit branch point in the plot.

The Scéalextric story-grammar does not permit repetition of the same action within the same plot, and so each action is unique, and denotes a unique juncture within each story. However, the current state of the narrative is more than the current action, and must include all the expectations that we carry into it from past events. Those expectations determine the extent to which the current action is surprising, and the extent to which characters feel shocked or disappointed by unfolding events. We might, for instance, expect characters that are shocked by the current action to react with greater emotional force than ones who see it as a natural outcome.

We must first characterize the emotions arising from a single action, for both its agent and the patient it affects. We can then quantify the halo of emotions that carry over from earlier states and add to the current emotional load. The range of emotions that we can distinguish – from a set of basic or pure emotions to complex blends of these primary colours – is large (Ortony, Clore, and Collins 1988), but a survey of the 800 plot actions suggests eight that are most useful: respect, disrespect, inspiration, disappointment, support, aggression, attraction and repulsion. Specifically, we want to quantify the degree to which one participant to an action may feel respected or disrespected, inspired or disappointed, supported or attacked, attracted or repelled, and calmed or aroused. We will quantify by degrees, on five parallel scales:

- disrespect++++
- disappointed+++  
- attacked+++    
- repelled+++    
- calmed+++      

For every action in the Scéalextric vocabulary, and for each A and B role of that action, we mark the expected position of the character filling the role on each of these five scales. For example, for the action A worshipped by B, we record that A feels respected+++ (the highest degree of respect) and B feels inspired+++ (the highest level of inspiration). Conversely, for the action A betrayed by B, we record that A feels supported-- (the lowest level of support) while B feels inspired-- (a low level of inspiration, but not the lowest). We do this on five scales for each role in all 800 actions. The arousal scale marks the intensity of each response. For instance, one is more aroused when one hates than when one dislikes, or when one worships rather than merely admires.

The first four dimensions are emotionally charged, since they mark out emotions with a positive or negative valence. The mean value of these dimensions thus provides the overall valence of an action from the perspective of a given actor. The fifth is not charged in this way – one can be as aroused by hate as by love – but it does signify the energy with which a feeling is experienced. Valence quantifies the impact of an action on an actor, while arousal suggests the scale of the actor’s dramatic response (Kensinger and Schacter 2006). A high-arousal feeling calls for a dramatic, high-energy gesture; a low-arousal feeling calls for a more subtle enactment.

To capture the emotional inertia of a character, we need more than the mean valence of their feelings at a given time. A weighted average of the aggregate valence of a character’s feelings from one action to the next – giving 50% weight to the current action – allows us to smoothly track changes in a character’s perspective over time. When this inertial valence undergoes a significant shift, to the positive (a sudden boon) or the negative (a sudden disappointment), this indicates a macro-level change that merits a macro-level interpretation.

In a sense, the horizontal scales mirror the physical stage, since each response can move actors closer together or further apart. A and B maintain an emotional distance to each other that grows or shrinks with each new plot event. While there is no quantitative difference between supported- - - and attacked+++ there is a qualitative one: the former reflects a failed expectation of support, while the latter incorporates no prior expectation of aggression. A character who is betrayed is not just figuratively attacked by another; they have their expectations of support dashed at the same time. Likewise, attacked-- reflects a lack of expected aggression and a high level of support, as when A surrenders to B. So this notation permits us to take a wholly quantitative view of the emotional effects of an action, while also allowing us to encapsulate a qualitative sense of our surprise at those effects. This will prove especially useful when we consider irony.
We can consider each parallel scale in isolation or in the aggregate. When considered in isolation, we can compare the current scalar settings to those of the previous action, to quantify the emotional shift wrought by the current action. When this current action comes as a surprise, a plot twist of sorts, we can expect significant jumps on some or all scales. Plot twists in *ScáleXtric* are woven into its story grammar, which also dictates the use of ‘but’ or ‘then’ to link actions. As such, its twists are purely formal products of disjunctive choice that are not interpreted emotionally by the generator. Yet, as the grammar tells the generator to insert but or then, an emotional layer can explain why this must be so, in terms of how the characters experience this turn of events. This is what it means for the system and its performers to interpret a story action in the light of past events. It is on the basis of this interpretation – the emotional shift from one event to another – that performers can choose to react figuratively, to obey the scripted norm or use metaphor or irony instead.

When should a performer choose a figurative response? Without interpretation, metaphor is just another disjunctive choice. With interpretation, a performer can reason that the scripted response is inadequate for the context in which the action is being performed. The standard script is inadequate when the feelings evoked in the moment are more intense than the current action, viewed in isolation, would suggest. Metaphor, or indeed irony, is an opportunity to recalibrate and fine tune the performer’s responses to suit the moment. Suppose A has shown favour to B in some way, perhaps by promoting B, or sharing a story with B, or confiding in B, and B responds by insulting A. Viewed in isolation, this act of repudiation should make A feel quite disrespected (+++), and even somewhat attacked (+). The dialogue model will suggest a scripted response to A that mirrors these feelings. However, when seen in the context of its previous actions, A should feel even more disrespected (+++) and feel all the more attacked (++ or even +++). The most apt response for A, then, is not to act as if insulted, and to speak and gesture accordingly, but to react as though physically attacked. That is, A should speak the lines of an attack victim, and act out the gestures of one who is under attack (e.g., “Get off me!” spoken while extending the arms and stepping backwards).

Taken together, the words and gestures of A and B are no longer a performance of the single action A insulted by B. Rather, the result is a conceptual blend in words and actions (Fauconnier and Turner 2002): B acts out an insult while A acts out an attack. This captures the truth of the situation as seen by A, but it also produces a novel blend that adds diversity to the performance. Actions interpreted and enacted in context lead to more varied enactments than they otherwise would. An ironic response emerges in much the same way, though the ensuing blend is more granular. Consider an action, A stands up to B, that has A feeling uninspired (inspired–). While inspired– is comparable to disappointed++ in scalar terms, since each causes the same shift on the same scale, it also encodes a failure to be as inspired as one expects. Irony is a playful response to this failure of expectations, insofar as it signals the failure while pretending that it has not occurred, thus highlighting the gulf between expectation and reality (Garmendia 2018). To pretend in a performance context, a performer can invert the valence of the expectation, e.g., to obtain inspired++ from inspired–, and then act out an action associated with the inverted emotion.

For instance, A can ironically perform a bow down to B instead of a stand up to B, by actually bowing to B, while nonetheless speaking the lines scripted for A stand up to B. The clear friction between this dialogue (for example, “I’ve had enough of you!”) and the oddly respectful gesture tips the wink to the audience that all is not as it seems. We use the gestural channel to carry the irony because it is a much more suggestive, and far less direct, carrier of meaning.

These approaches to irony and metaphor are compatible, and play well together to create an ironic overstatement. For instance, if metaphor is used to map A bow down to B onto A worship B, to exaggerate the extent to which A looks up to B, the irony of acting out the supplicant gestures of A worship B while vocalizing A stand up to B is sharper still. This remains a relatively safe means of incorporating irony into a performance, as the spoken dialogue keeps the action moored to the literal basis of the story. Gestures are often more subtle than words, and offer more plasticity to an actor.

Robots offer quirkier modes of expression, such as flashing LEDs, that are less intuitive than either words or gestures (Häring, Bee, and André 2011). Indeed, gestures are a vital part of embodied communication (McNeill 1992), and many have the same deep, conceptual roots as words. Still, a fixed set of task-specific gestures is often defined for robotic performances (Wilcock and Jokinen 2013; Csapo et al. 2012), but even these bespoke gestures are subject to cultural variability. Speakers of the Aymara language, for instance, refer to future events not by pointing ahead but by pointing behind (Núñez and Sweetser 2006). It pays, therefore, as far as it is practicable, to rely less on iconic gestures that are culturally rooted (such as kneeling to propose) and more on schematic movements that have a greater claim to universality.

A performer’s gestures should always rhyme with their words, unless irony is used to create a knowing dissonance. The approach to metaphor and irony presented here works primarily at the conceptual level of plot actions, and so the choice of words and gestures follows from this. Metaphors arise when a plot action is mapped to another, semantically similar action that more intensely evokes a certain emotion. This intensification explains why metaphor is an asymmetric form of comparison that marries similarity to directionality. Conversely, irony is achieved when one action is mapped to a semantic opposite that evokes the inverse of an emotion. So, irony relies on opposition rather than similarity, and on inversion of emotions rather than their intensification. Still, a mix of irony and metaphor can use opposition, similarity, inversion and intensification to produce a satirical effect.

In each case, however, after a mapping between actions is achieved at the plot level, the corresponding dialogue and gestures are chosen because they happen to be associated a priori with the given actions. While these purely disjunctive choices are driven by emotional choices at the action level, they can also be grounded in an emotional interpretation. Consider how a gesture is chosen to enact a specific action. Many gestures are pantomimic and culture-specific, but other movements, such as relative motion between per-
formers, are more subtle and less dramatic, but just as communicative (Wicke and Veale 2021). The arbitrariness of many gestures in cultural and dramatic terms makes it appealing to simply define gestures as black-box scripts for different actions. But we can also annotate gestures on an emotional level, using the same emotion scales that are employed for plot actions.

For example, the bowing gesture is now annotated with \textit{respected++} for the actor in front of which it is performed, and annotated with \textit{inspired+} for the actor that performs it. It is vital that we separate each gesture from the action that it may embody, so as to annotate each one on its own terms. This ensures that the emotions we associate with a gesture actually reflect the audience’s reactions to this gesture, and not our desired reaction to the action it means to enact. As this is a tricky knot to unpick, we plan to crowd-source the emotional annotations for each gesture in an empty context. For the present, we annotate each gesture ourselves, so that the performers can use these annotations to select the most appropriate gesture to perform for any given action. In this way, robot performers can choose the gestures that reflect the emotions of a scene as they interpret them in context.

\textbf{Opportunities for Metaphor and Irony}

Valence and arousal are key elements of suspense (Delatorre et al. 2016). They enable us to keep listeners on the edge of their seats before arouses them with a sudden plot twist. At key points, the inertial valence of a character’s perspective can flip from positive to negative, or vice versa. When a substantial shift in inertial valence occurs at the current action, it will have been brewing for some time. When this shift exceeds a fixed threshold $\Delta$, an actor might mark the shift with an exaggerated response that goes beyond the script. Fig. 2, which tracks the inertial valence of characters that fill the A (blue line) and B (orange line) roles in a \textit{Scéalxtric} plot of 31 actions, highlights shifts that support metaphor and irony.

These shifts are marked by vertical lines in Fig. 2: dashed lines mark those supporting a metaphorical response, dotted lines mark those that support an ironic response. The plot in Fig. 2 offers three shifts where irony is supported (upper annotations) and four where metaphor is a strong possibility (lower annotations). Irony is a valid response when the fall in a character’s inertial valence exceeds the threshold $\Delta$, signifying a failure of expectations. Irony is also supported when an action is explicitly tagged with a failed expectation such as \textit{respected- -}. This is equivalent in scalar terms to \textit{disrespected++}, yet it does more than convey disrespect; it also captures a failure to be as respected as one expects. An actor can now make an interpretative choice to show \textit{respected- -} with a gesture that instead implies \textit{respected++}, while relying on narration and dialogue to make the disrespect clear.

A metaphorical response is appropriate when a character’s inertial valence is greater (by $\Delta$ or more) than the specific valence of the current action, and there is a need to dramatize this lagging emotional load. This dramatization is achieved by enacting a different but similar action with a valence (for that character) that is closer to its inertial valence in context. For instance, at plot point #10 in Fig. 2, an actor might react with gestures that suggest a character is despised (\textit{disrespected++}) rather than merely resented (\textit{disrespected++}).

We estimate the opportunities for the performers to make interpretative choices by analyzing 10,000 generated stories with a setting of $\Delta = 2$ (so a character’s inertial valence

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig2.png}
\caption{Interpretative decision points in a story with 31 plot actions. Annotations at top explain ironic interpretations. Boxes at bottom explain metaphorical exaggerations. Blue lines track the inertial valence of character A; orange lines track that of B.}
\end{figure}
must shift by 2 points or more to enable irony or metaphor).
In this sample of 10,000 stories, 47,532 interpretative choice points are identified: 34,810 support metaphor, and 22,352 support irony. At this setting, a story has 4 to 5 opportunities for interpretative choice; 3 to 4 for metaphor and 1 or 2 for irony. We can adjust the $\Delta$ parameter to allow for more or less interpretation by the actors. Thus, when $\Delta = 1$ the average story provides 10 opportunities for interpretative choice, 8 to 9 for metaphor and 3 to 4 for irony (the two are not mutually exclusive). When $\Delta = 3$, the average story offers just 1 or 2 opportunities for interpretative departures from the script. The ideal setting for $\Delta$, i.e. the setting at which audiences are happiest with the performers’ use of metaphor and irony, has yet to be experimentally validated in user studies.

Evaluating Action At A Distance
Most gestures are ephemeral and transient, performed once and quickly forgotten. Few, if any, are persistent from one action to the next. While a performer can rely on gestures to express the emotions of the moment, they do not capture the inertia of the story so far. Actions that do not persist are not summative, and so do not reflect an inertial view. Ideally, if gestures are to be chosen on the basis of an interpretation of story events, some should also reflect this interpretation.

However, there is a class of physical actions that reflects a summative view of a story and its character interrelations at a given time: spatial movement to and fro on a stage. We noted earlier that some actions evoke emotions that move characters closer together or further apart in abstract terms, and this metaphorical movement is easily translated into the relative physical movement of robot performers. Motion of this kind is persistent: if a performer steps forward or backwards, the performer takes a step forward; if a negative shift exceeds $-\Delta$ they take a step backwards. Performers move as their current interpretations dictate, while leaving a gap that offers a more global perspective. This is how space is used coherently to convey meaning and emotion.

This hypothesis has been validated using crowd-sourced user studies (Wicke and Veale 2021). In the incoherent condition, the robot performers do the opposite of what their spatial logic dictates. Conversely, the coherent condition has them follow this logic to the letter. We also conduct an evaluation in which it is a robot’s gestures that are chosen coherently (to suit the action) or incoherently (at random). In each condition of each evaluation, judges are shown recorded fragments of a performance of the same Scéalextric story.

Methods The evaluations are crowd-sourced using AMT, Amazon’s Mechanical Turk. Short 1-minute videos of robot performances, focusing on just two story actions, are shown to participants. Our pilot studies show that this format emphasises the movements of the robots and keeps participants engaged. A pool of $N = 160$ volunteers is evenly divided into four conditions: coherent space, incoherent space, coherent gesture and incoherent gesture. Each rater answers 14 questions about a video relating to just one condition. The first 7 measure the perceived attractiveness of the performance overall, while the second 7 relate to aspects of the embodied performance (e.g. whether the robots appear natural, or whether the participant would like to see the whole story). In addition, each questionnaire contains extra gold standard questions to weed out those who do not engage.

Analysis We excluded 42 participants for failing the gold-standard tests. The 118 valid responses are distributed across the four conditions as follows: coherent space (32), incoherent space (29), coherent gesture (29) and incoherent gesture (28). When each 14-item questionnaire is aggregated to yield a single appreciation score, a two-way ANOVA for coherence versus incoherence and space versus gestures shows a significant preference for the coherent performances (mean squares = 48.138, F values = 16.147 and p value <0.001). In particular, a significant difference is observed between the coherent space and incoherent space conditions (Bonferroni corrected p value = 0.047). Moreover, a post-hoc t-test shows significant differences between the coherent and incoherent conditions, while the effect size of Cohen’s $D = 0.197$ indicates a small to medium effect that favours the coherent conditions. Conversely, since no significant differences between the spatial and gestural conditions are observed, it appears that audiences appreciate one as much as the other.

Results The results indicate that audiences appreciate performances that deliberately make interpretative use of space. Moreover, this schematic and logically simple use of space is just as effective as the use of a great many ad-hoc gestures.

Concluding Remarks
Some key distinctions in computational creativity invite a binary perspective when much greater nuance and gradation are called for. Consider, for instance, Boden’s distinction between P- and H-Creativity (Boden 1990), which separates innovations that are original in a historical sense from those that merely seem novel to their producer. If novelty is instead judged on a graded scale, artifacts can be seen as more or less H- or P-Creative, with many exhibiting an affinity to both poles at once. Mere generation is yet another distinction that invites a binary perspective, and one that also reveals itself as graded on closer inspection (Ventura 2016).

Indeed, applying as it does to a process as a whole, the distinction appears so binary and so judgmental that it no longer seems fit for purpose. As a replacement, we instead propose purely distinctive choice (or pure disjunction), since this affords greater nuance and greater scope for gradation. A process or a system can employ pure disjunction for one decision and interpretative choice for another, and so should be judged on how it achieves a balance of these alternatives.
Pure disjunction remains an attractive option for many generative tasks, and especially so for systems that aim to surprise with meagre resources. Unless interpretative choice is designed to surprise, its natural tendency is to tame the wildest excesses of random selection. Interpretation makes concessions to what it is familiar and normative, so as to couch the novel in the expected and achieve an "optimal innovation" (Giora et al. 2004). If too many concessions are made – as e.g., regarding gender roles in stories – interpretation bolsters the status quo. By paying little regard to what is normative in a relationship, a family or a workplace, pure disjunction can shatter norms to yield transgressive results.

As we must carefully balance interpretative and disjunctive choice to achieve the desired mix of novelty and value, we have presented a measure of inertial valence to dictate when a generative system should make the shift from one to the other. This measure is, in a sense, an “objective correlata” as defined by the poet T.S. Eliot (Barry 2002), allowing an actor’s most dramatic choices to be adequately rooted in their character’s unfolding relation to the underlying text. As such, it is instructive to note that non-computational practitioners, such as poets and critics, have also defined objective functions of their own, and computational creativity is a logical, computational extension of those critical approaches.

References


Conceptual Expansion Neural Architecture Search (CENAS)

Mohan Singamsetti¹,², Anmol Mahajan¹,², and Matthew Guzdial¹,²
¹Computing Science Department, University of Alberta
²Alberta Machine Intelligence Institute (Amii)
{singamse, mahajan, guzdial}@ualberta.ca

Abstract

Architecture search optimizes the structure of a neural network for some task instead of relying on manual authoring. However, it is slow, as each potential architecture is typically trained from scratch. In this paper we present an approach called Conceptual Expansion Neural Architecture Search (CENAS) that combines a sample-efficient, computational creativity-inspired transfer learning approach with neural architecture search. This approach finds models faster than naive architecture search via transferring existing weights to approximate the parameters of the new model. It outperforms standard transfer learning by allowing for the addition of features instead of only modifying existing features. We demonstrate that our approach outperforms standard neural architecture search and transfer learning methods in terms of efficiency, performance, and parameter counts on a variety of transfer learning tasks.

Introduction

Deep learning is the study of deep neural networks (DNNs), which are a type of function approximators. DNNs have achieved remarkable success in various challenging applications such as image classification, image generation, and natural language processing (Szegedy et al. 2015). Modern deep learning approaches perform well when researchers train large models, often with at least millions of parameters, on large amounts of data (Halevy, Norvig, and Pereira 2009; Brown et al. 2020). However, this leads to two problems. First, the size of these models limits where they can be applied and who can afford to train them. Second, there are application domains in which we do not have sufficient training data, and therefore where we cannot currently apply these approaches.

The limitations of large pre-authored architectures have been addressed using Neural Architecture Search (NAS). In this approach, the model architecture is optimized along with the model weights. NAS has outperformed manually designed architectures in some tasks such as object detection (Zoph et al. 2018) and image classification (Zoph et al. 2018). However, NAS is a time consuming and computationally expensive process (Li and Talwalkar 2019) since it requires training many potential architectures from scratch.

Approaches exist to transfer the knowledge of models trained on large source datasets to tasks with smaller target datasets, including transfer learning (Lampert, Nickisch, and Harmeling 2009), domain adaptation (Daumé III 2009), few-shot learning (Fei-Fei, Fergus, and Perona 2006) and zero-shot learning (Xian, Schiele, and Akata 2017). However, these approaches all require re-training the models (Levy and Markovitch 2012), or require manually authoring or learning secondary features (Xian, Schiele, and Akata 2017) to handle new cases or to adapt to a new domain. In addition, these approaches generally assume fixed architectures set by a human expert according to the target task. This is a limiting factor, as the transfer learning process can be forced to adapt knowledge rather than retain it due to the fixed size. For example, when attempting to adapt a source network trained to recognize cats to a target network to recognize dogs some features would be better to retain (e.g. fur), while others would be better to replace (e.g. cat eyes).

Intuitively, if we could combine NAS and transfer learning we could end up with an approach that could find more efficient models more quickly, even for cases with less data. NAS could benefit from transfer learning as weights can be transferred from existing networks, speeding up the process of training novel architectures. Transfer learning could benefit from NAS as it can allow for the addition of new features instead of only the modification of existing features learned from a source dataset. We believe this has the potential to lead to smaller and more accurate models that can be trained more quickly, and therefore represents a valuable open problem. Beyond recent domain adaptation and architecture search work (Li and Peng 2020), this combination of neural architecture search and meta learning is a largely unexplored area of research. One of the reasons for this may be that transfer learning often represents a much faster and simpler optimization problem than training from scratch. In comparison to this, naive architecture search represents an unbounded search problem.

Combinational creativity (Boden and others 2004), also called conceptual combination (Gagné and Shoben 1997), is a cognitive process in which old knowledge is combined to produce new knowledge. This is a general process in human cognition (Gagné and Shoben 1997; Fauconnier 2001), an efficient means of representing new concepts with existing knowledge. Attempts have been made to approximate
this cognitive process computationally, most famously conceptual blending, to the point where any attempt to approximate combinational creativity with computation is called blending (Fauconnier 2001). However, these earlier combinational creativity approaches are generally limited to hand-authored inputs and curated knowledge bases.

Our problem in this paper directly relates to combinational creativity (Boden and others 2004). In combinational creativity, existing knowledge is recombined to create new knowledge. As a cognitive process, it is unlikely that our human brains replicate the existing knowledge in order to produce a recombination, as this would be inefficient and slow. Instead, evidence suggests this is a quick, compact process (Gagné and Shoben 1997). We therefore argue that simultaneous neural architecture search and transfer learning is a reasonable computational metaphor for combinational creativity in neural networks.

To address our problem we employ a representation that allows for sample-efficient transfer learning called Conceptual Expansion (Guzdial and Riedl 2019; Banerjee 2021; Guzdial and Riedl 2021). In this transfer learning approach the reuse of a source model’s knowledge is modeled as a combinational creativity problem. With conceptual expansion we can approximate the weights of a target model as a combination of weights from a source model. This allows for a much faster optimization of model weights, therefore speeding up architecture search. We call our approach Conceptual Expansion Neural Architecture Search (CENAS). In a number of image classification domains we demonstrate how CENAS outperforms standard architecture search, transfer learning, and naive architecture search with transfer learning. This work contributes this novel approach, experimental results that demonstrate that it outperforms existing approaches to meta learning and architecture search (Li and Peng 2020), and earlier applications of conceptual expansion to deep neural networks (Guzdial and Riedl 2019; Banerjee 2021).

Related Work

In this section we overview the two most related areas of prior work: architecture search and transfer learning.

Architecture Search

Architecture search attempts to automatically determine the optimal neural network architecture for a particular problem. The approach dates back to the 1980’s, when evolutionary optimization approaches were proposed to find both the architectures and weights of a neural network (Miller, Todd, and Hegde 1989). As the name implies, the problem is typically represented as a search problem, where some initial architecture or population of architectures are optimized to find the best architecture in a given search space (Xie and Yuille 2017). We employ evolutionary search in this work as it has been shown to still be the best or equivalent optimization method for neural architecture search (Real et al. 2019). Many optimization strategies have been used to explore the space of the possible architectures (Zoph and Le 2016). However, architecture search methods still struggle to find the same results as architectures hand-authored by human experts.

Transfer Learning

Transfer learning of deep neural networks (DNN) refers to the transfer of knowledge from a DNN trained to solve one source problem to a DNN designed to solve a related target problem. A wide range of prior approaches exist for the transfer of knowledge in neural networks such as domain adaptation and one or zero-shot learning (Fei-Fei, Fergus, and Perona 2006; Xian, Schiele, and Akata 2017). These kinds of approaches often require additional features to guide the transfer of knowledge, which can be hand-authored or learned from a secondary dataset (Ganin et al. 2016). Our approach does not use any additional hand-authored or machine-learned features.

Domain adaptation focuses on transferring the knowledge gained from one or more labelled domains to an unlabelled target domain. Multi Source Domain Adaptation (MSDA) takes training data collected from multiple sources and applies that to a single unlabelled target (Peng et al. 2018). Neural Architecture Search for Domain Adaptation (NASDA) is a recent approach focused on deriving the best architecture for a specific domain adaptation task by leveraging differentiable neural architecture search (Li and Peng 2020). Our approach also uses a unique representation similar to NASDA, but it is a representation that approximates novel class features in the target domain as combinations of source domain features.

Combinational creativity (Boden and others 2004) or conceptual combination (Gagné and Shoben 1997) represents the ability of humans to combine existing knowledge to produce new knowledge. Computational implementations of combinational creativity can be understood as a specialized case of transfer learning, focused on re-representing existing knowledge to approximate new or unseen concepts. There have been many combinational creativity approaches with conceptual blending being the most popular (Fauconnier 2001; Guzdial and Riedl 2018). However, the majority of these existing approaches can only take hand-authored symbolic data as input. Guzdial and Riedl introduced combinets (Guzdial and Riedl 2019), the application of combinational creativity to deep neural networks via a representation they called Conceptual Expansion, which is designed to work with messy, machine-learned knowledge. We build directly upon this work, but extend it to a more general approach that better leverages the available, existing data to do simultaneous transfer learning and architecture search.

System Overview

In this section we present our Conceptual Expansion Neural Architecture Search (CENAS) approach. This approach is focused on domain transfer problems: where we have distinct source and target datasets, and where the goal is to adapt knowledge from the source domain to the target domain. We define CENAS as a three-step process. First, we train a model on the source dataset. Second, we approximate the weights of the connections of an initial target model as
a conceptual expansion: a combination of weights from the source model. Third, we run our architecture search process on target model by updating our approximations of existing connections and approximating the weights of any additional connections in the same manner. This process is visualized in Figure 1.

**Conceptual Expansion**

We begin by formally defining Conceptual Expansion (CE). CE is a way of representing knowledge as a combination of existing knowledge, in our case neural network weights. If we wished to represent a particular weight \( w \) as a combination of existing weights with CE we would use Equation 1:

\[
CE^w(F, A) = a_1 * f_1 + a_2 * f_2 + \ldots + a_n * f_n
\]

where the \( F = f_1, f_2, \ldots, f_n \) represent a set of existing weights, \( A = a_1, a_2, \ldots, a_n \) are alpha filter matrices that undergo pairwise multiplications with their paired existing weight matrix. \( A \) acts as instructions for how to transform the weight for the combination. Notably, the same weight value can appear multiple times in \( F \) with different \( a \) values. CE can be understood as analogous to the crossover function in evolutionary search (Whitley 1994), both rely on the intuition that combinations of high-quality knowledge are more likely to lead to new high-quality knowledge. Similarly, crossover can be understood as another example of combinational creativity, the general human cognitive process for reusing old knowledge (Gagné and Shoben 1997; Fauconnier 2001). CE is designed to be a simple but extendable way to represent combinations of neural network weights, in order to more clearly study combinational creativity in DNNs. In the original CE paper (Guzdial and Riedl 2019), Guzdial and Riedl employed greedy search to optimize the \( f \) and \( a \) values, finding that this approach outperformed backpropagation-based transfer learning approaches for low sample sizes. The values in \( a \) matrix range [-2, 2]. However, the greedy optimization failed to outperform existing methods for larger sample sizes of data.

**Source Training**

In this paper we focus on the image classification domain, employing several common image classification datasets as our source and target datasets. We employ CifarNet as our initial source model throughout this paper (Krizhevsky, Hinton, and others 2009), as it represents a well-understood and compact initial architecture. CifarNet has two convolutional layers each with max pooling and two fully connected layers. We implemented it unchanged from the original description (Krizhevsky, Hinton, and others 2009). The convolutional layers apply the convolution operation to the input in order to extract features. Max Pooling is a convolution process where it down-samples the feature representation by taking only the maximum values. Our first step is to train CifarNet on the source dataset.

**Weights Approximation**

The second step of our approach approximates the weights on an initial target model. Given that we focus on image classification in this paper, we only need to approximate novel weights for the final classification layer in this step. To approximate these weights we take our available training data for each target class and pass it through the model trained on the source dataset. This gives us a distribution over the \( n \) source classes for each target class (e.g. In a target domain “fox” class images might be classified as the source domain class “dog” 65% of the time and the remaining 35% of the time as “cat”). We normalize these values with a softmax function, which gives us our initial \( a \) values. Softmax is an activation function which maps the output in the range \([0, 1]\) and also maps each output in such a way that the total sum is 1. Every one of the non-zero values output from the softmax function is paired with its source \( w \) value and combined to approximate the initial target class \( w \) value as in Equation 1 (e.g. the source domain class cat and dog weights are associated with their alpha values to approximate the initial target domain class fox classification neuron weight). For all other layers in the model we initialize their \( a \) value as a matrix of ones of the appropriate shape for that layer’s weights. Thus the initial model is represented as a conceptual expansion that is equivalent to the source model except for the final layer.

**Architecture Search**

For the architecture search step of CENAS we use an evolutionary optimization process or genetic algorithm, given their history and consistent performance in NAS tasks (Real et al. 2019). We chose this as greedy optimization of CE struggles to exit the local optima near the model output from the weight approximation step (Guzdial and Riedl 2019; Banerjee 2021). Evolutionary optimization requires that we initialize a population of points, define mutation and crossover operators, and a fitness function. We represent this whole process in Algorithm 1, with lines one and two representing the first and second steps of CENAS described...
We employ a total of seven different mutation operators. Our first four operators are typical for architecture search applications:

- The first mutation operation adds a new convolutional layer to the architecture at a random position before fully connected layers and after the first convolutional layer. We randomly choose 32 filters or 64 filters and a kernel shape of 3x3 or 5x5, we use all the other parameters from the original CifarNet’s convolutional layers.
- The second mutation operation deletes a random convolutional layer in the network besides the first one to maintain the fixed input size.
- The third mutation operation adds additional filters to an existing convolutional layer. We randomly choose a convolutional layer in the network except for the first layer and add a random filter count of 2, 4, 8, 16, or 32.
- The fourth mutation operation deletes filters in a random convolutional layer of the network; we delete a random filter count of 2, 4, 8, 16, or 32 filters.

The remaining three mutation operations help in manipulating the network weights directly by modifying the $a$ and $f$ values associated with each weight.

- The fifth mutation operation multiplies the $a$ of a random $f$ of a random layer by a scalar in the range [-2,2].
- The sixth mutation operation replaces a $f$ value of a random layer with a randomly selected $f$ value.
- The seventh mutation operation adds a random $a$ and $f$ to a random position in a random layer (e.g. adding $a_{n+1} f_{n+1} \rightarrow a_1 f_1 + a_2 f_2 + ... + a_n f_n$).

## Experiments

We focus on convolutional neural networks (CNNs) for our initial exploration of CENAS, as they were used in prior combiners (Guzdial and Riedl 2019; Banerjee 2021). We explore this through two major types of experiments in this paper. First, we measure the performance of CENAS on several tasks that have been used in prior domain transfer and architecture search work, alongside several baselines (Li and Peng 2020). This is meant to present evidence for our claims of the value of CENAS to transfer and architecture search tasks. Second, we present a series of low data $n \rightarrow n+1$ tasks using the CIFAR-10 (Krizhevsky, Hinton, and others 2009) dataset. This is meant to present evidence for our claims around how CENAS can operate even for low data problems, which relates to how humans can employ combinational creativity with few examples (Gagné and Shoben 1997). In addition, this second evaluation allows us to directly compare to the original CE with DNNs work (Guzdial and Riedl 2019).

For all tasks we make use of CifarNet as our base architecture. We train CifarNet for 100 epochs on our source dataset. For CENAS we then use the the training dataset as described above to guide the search over models for 100 generations with a population of size 10. We chose these low values to demonstrate the effectiveness of the approach with minimal computation, and as part of our investigation as to whether

---

**Algorithm 1: CENAS Workflow**

**Input:** An architecture $A$, the population size $\text{pop.size}$, maximal generations $\text{gen}$, the source dataset, and the target dataset.

**Output:** Best performing architecture.

1. $A \leftarrow \text{train } A \text{ on source;}$
2. $A \leftarrow \text{Re-represent } A \text{ using CE and target}$
3. $\text{pop} = \{ A \}$
4. while $|\text{pop}| < \text{pop.size}$ do
5.   network $\leftarrow \text{Mutation}(A)$
6.   $\text{pop.append}(\text{network})$
7. end
8. $i \leftarrow 0$
9. while $i < \text{gen}$ do
10.  $\text{pop} \leftarrow \text{Crossover}(\text{pop})$
11.  $\text{pop} \leftarrow \text{Mutation}(\text{pop}, \text{mutationRate})$
12.  $\text{fitness}_\text{pop} \leftarrow \text{Fitness}(\text{pop})$
13.  $\text{pop} \leftarrow \text{Reduce}(\text{pop}, \text{fitness}_\text{pop})$
14.  $i \leftarrow i + 1$
15. end
16. architecture $\leftarrow \text{best_model}($pop$)$
17. Return architecture;

**Algorithm 2: Fitness Score**

**Input:** An architecture $Net$, target domain data $D_{target}$ with $N$ classes of image dataset, $target\_classes$ are list of classes in target domain

**Output:** Fitness scores of $Net$

1. $\text{score} \leftarrow 0$
2. for $\text{class in target\_classes}$ do
3.   $\text{score} \leftarrow \text{score} + \text{Net.accuracy}(D_{target}[\text{class}])$
4. end
5. Return $\text{score}/N$;

above, and the remaining lines devoted to this final step. We initialize a population of fixed size based on the output of the second step, running our mutation function $\text{pop.size}$-1 times to produce each population member. From there we iterate through the standard evolutionary search steps, running our crossover function to double our population size, mutating the population members, evaluating the new models on the target domain training data, and reducing back to our original population size. The mutation and crossover functions act directly on the model architecture and weights; we describe them in detail below. We explored several fitness functions, but found that taking the average accuracy over the target domain training data gave the best performance as represented in Algorithm 2.

**Mutation and Crossover**

For our crossover function we use a simple single-point crossover. We take two models and target a random CNN layer from each for the split point. We then take the first half of one model and the second half of the other.
this can be an appropriate metaphor for combinational creativity in neural networks. We take the final 10 members of the final generation and train them for 30 epochs on our target dataset in a standard supervised learning paradigm using RMSProp. In all cases we use a batch size of 32 and a learning rate of 0.0001. We used Keras for implementation and non-CENAS training of our deep neural networks.

We have four baselines. The first two are variations of CENAS. R-CENAS employs a random walk instead of a genetic algorithm. It uses the same seven mutation functions from above and chooses one at random for every architecture. We run the random walk for 100 steps and output the top five best models according to target training accuracy. G-CENAS makes use of greedy optimization instead of a genetic algorithm. We try 10 random mutation functions at every step as a neighbor function and choose the best across the neighbors and current model according to training accuracy. We run for 100 iterations and take the final five models as our output. Afterwards, we take this output and train it for 30 epochs using the target training data.

The next two methods represent how one might naively attempt to solve this problem using more standard methods. The first of these is a simple neural architecture search implementation (NAS). For this implementation we used the same fitness and crossover functions from our CENAS implementation, but only the first four mutation functions, making it a more standard NAS implementation. After mutation and crossover functions we instantiate the new model and train on the available target training data for 30 epochs. If this naive NAS outperforms CENAS it would indicate that our approach to transferring existing features via recombination is actively detrimental. Finally, we include a naive combination of neural architecture search and transfer learning (NAS-T). This is similar to our naive NAS implementation but we transfer existing weights from the parent models during crossover and copy over the weights from the most similar weight or filter for our mutation functions. We then train on the target training data for 12 epochs, as we found that any more training led to overfitting. If NAS-T outperforms CENAS that indicates that our more complex representation has no benefit over simply finetuning the existing weights, and is therefore no better as a metaphor for human combinational creativity. For all of the NAS approaches (NAS, NAS-T, and CENAS) that rely on a genetic algorithm we report the results for the top five members of the final generation according to training accuracy.

All the experiments are carried out using the cloud computing resources of Compute Canada, which uses 32 cores 4 x NVIDIA V100 Volta (32G HBM2 memory). We employ a consistent random seed across all experiments.

Domain Transfer Experiments

Setup We make use of four tasks and datasets inspired from prior domain adaptation work (Hoffman et al. 2018; Li and Peng 2020). The four datasets are MNIST, USPS, STL-10 (Coates, Ng, and Lee 2011), and CIFAR-10 (Krizhevsky, Hinton, and others 2009), which are all well-understood image classification datasets. Each of our tasks involves using one of the datasets as a source and another as a target, making our four tasks: MNIST→USPS, USPS→MNIST, STL-10→CIFAR-10, and CIFAR-10→STL-10. Given that CifarNet was designed for CIFAR-10 we modify the other datasets to have the same 32×32 input size, but do not otherwise process them, unlike in prior domain adaptation work where certain classes are removed (Hoffman et al. 2018; Li and Peng 2020).

Results We present the average test accuracy and standard deviation for each target dataset using the given test splits, and the average parameter count of the output models for each approach in Table 1. As a comparison point, the default CifarNet has 597K parameters. Overall, CENAS outperformed the baselines on three of the four tasks. The only task it struggled on was the MNIST→USPS task, which seems to be a difficult domain transfer task given prior results (Li and Peng 2020). We anticipate this is due to transferring from a monochrome to an RGB colour domain. The NAS approaches that involved backpropagation to a greater extent were able to better adapt to this new domain. However, while our final CENAS models were roughly 5% less accurate they were also three times smaller. Of particular interest are the USPS→MNIST results, which outperform even supervised domain transfer approaches reported in prior work (Hoffman et al. 2018).

While we do not include the results in the table or describe them as baselines, the very first model of the first generation for NAS and NAS-T represent training CifarNet on the target domain from scratch and finetuning CifarNet trained on the source domain respectively. Our finetuned CifarNet test accuracy (36.07, 99.37, 78.49, 77.64 on the four tasks) outperformed several baselines, but didn’t outperform CENAS. Thus, CENAS seems to be a better metaphor for the human ability to adapt to new knowledge by reusing existing knowledge than standard transfer learning. Comparatively, CifarNet trained only from scratch on the target domain with no adaptation or transfer outperformed CENAS in all but the third task (96.41, 99.52, 81.60, 78.51). This is still a positive result overall as NAS architectures tend to struggle to even equal the performance of hand-authored architectures (Saxena and Verbeek 2016). But another way, CENAS seems to be the best approach for automatically adapting to new knowledge, but not for learning knowledge directly.

In terms of average parameter count, CENAS is a clear winner, with hundreds of thousands of parameters fewer than the closest approach. This may seem unintuitive, but a similar effect is seen with network pruning (Liu et al. 2018) where reducing the number of weights can be beneficial as it leads to more general models. Notably, the fitness function did not bias the output towards smaller models, only accuracy. The models in the final iteration are also generally larger than the initial architecture. Instead, compactness arose as a secondary effect of pursuing accuracy and the combinational representation. These compact representations parallel prior cognitive science results in terms of the representative power of combinations (Gagné and Shoben 1997; Fauconnier 2001). This provides some evidence that this may be true in deep neural networks as well. Of particular note is the relative size to the relative performance of...
Table 1: Domain Transfer Tasks Average Accuracy and Parameter Count Results

<table>
<thead>
<tr>
<th>Approach</th>
<th>MNIST -&gt; USPS</th>
<th>USPS -&gt; MNIST</th>
<th>STL -&gt; CIFAR-10</th>
<th>CIFAR-10 -&gt; STL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>test acc</td>
<td>model para</td>
<td>test acc</td>
<td>model para</td>
</tr>
<tr>
<td>R-CENAS</td>
<td>74.9 ± 15</td>
<td>1.8M ± 9.9k</td>
<td>99.3 ± 0.0</td>
<td>2.1M ± 846k</td>
</tr>
<tr>
<td>G-CENAS</td>
<td>90.0 ± 8.3</td>
<td>903k ± 12.2k</td>
<td>99.3 ± 0.1</td>
<td>1.7M ± 615k</td>
</tr>
<tr>
<td>NAS</td>
<td>83.5 ± 12</td>
<td>1.5M ± 55.6k</td>
<td>99.3 ± 0.1</td>
<td>1.3M ± 539k</td>
</tr>
<tr>
<td>NAS-T</td>
<td>95.5 ± 2.4</td>
<td>1.6M ± 13.8k</td>
<td>99.2 ± 0.1</td>
<td>1.5M ± 13.6k</td>
</tr>
<tr>
<td>CENAS</td>
<td>89.94 ± 12</td>
<td>579k ± 834k</td>
<td>99.4 ± 0.1</td>
<td>920k ± 511k</td>
</tr>
<tr>
<td></td>
<td>82.5 ± 0.6</td>
<td>2.2M ± 170k</td>
<td>78.2 ± 0.1</td>
<td>2.18M ± 15M</td>
</tr>
</tbody>
</table>

Table 2: Domain Transfer Tasks Average Computation Time in GPU Hours/Days

<table>
<thead>
<tr>
<th>Approach</th>
<th>Average Hours (GPU Hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-CENAS</td>
<td>8H 36M</td>
</tr>
<tr>
<td>G-CENAS</td>
<td>6H 55M</td>
</tr>
<tr>
<td>NAS</td>
<td>2D 9H</td>
</tr>
<tr>
<td>NAS-T</td>
<td>2D 6H</td>
</tr>
<tr>
<td>CENAS</td>
<td>7H 23M</td>
</tr>
</tbody>
</table>

these models, with some of these final models performing comparatively to models up to 3-4 times their size (Kabir et al. 2020). Further, while CENAS is weakly supervised up to its final generation, prior unsupervised domain adaptation methods for these tasks far exceeded these parameter counts (Li and Peng 2020).

We present the average computation time in GPU hours in Table 2. R-CENAS was slightly slower than CENAS as the mutation functions occurred at every step, instead of only with some probability. While G-CENAS was on average somewhat faster due to early convergence, it’s clear from the average parameters of its output models that it was biased towards adding features. The big difference is between the approaches that were strongly supervised, that re-trained at every generation: NAS and NAS-T. While NAS-T was somewhat faster as it trained for less time to adapt the existing features, our CENAS approaches were, on average, three times faster than these methods.

10 to 500 samples of the held-out \((n + 1)\) or Novel class. To ensure our results were reproducible, we made use of the first \(X\) training instances in each experiment where \(X\) is the sample size of the training images of the Novel class. Notably we did not make use of backpropagation for the CENAS approach for this experiment, as it was not used in the original paper.

**Results** We visualize the results for the \(X=100-400\) cases per novel class in Figure 2. We only include the NAS-T baseline due to space constraints, and as NAS performed equivalently or better to NAS-T for these sample sizes. We also note that our experiments included a transfer-only (no architecture search) baseline, which performed significantly worse. The dotted line across all the graphs represents 85%, which is the reported CIFAR-10 test accuracy for CifarNet, though we only observed values closer to 80% when training on all available data (Krizhevsky, Hinton, and others 2009). From the Figure 2 it’s clear that CENAS outperforms the baselines when it comes to the held out or novel class at lower sample sizes. We found that NAS-T and NAS began to perform equivalently or better than CENAS without backpropagation at and above the \(X=400\) case. However, they had the same drawbacks as above in terms of model size and computation time.

We found that the accuracy on the held out class dropped to nearly 0 for all of our baselines for the \(X<100\) case. We visualize \(X=10-90\) sample sizes for CENAS separately in Figure 3. Each line indicates the average across the novel classes of the novel \((n + 1)\) and other \((n)\) classes. Interestingly, CENAS retains better than chance accuracy on the held out class (>10%) all the way down to 10 samples. These results mirror the earlier results with greedy optimization and without architecture search (Guzdial and Riedl 2019), and outperform follow-up work using other optimization methods (Banerjee 2021).

Interestingly, CENAS’ performance on the held out class does not correlate to the sample size. We hypothesize that instead of training data size, a secondary feature of the held-out class training set is more important: the extent to which it reflects the true variance of the class in question. We also anticipate that this is closer to human combinational creativity, though we are unaware of prior work that investigates this. If this is true, it could lead to even stronger results with tailored datasets. We hope to study this in future work.

**Limitations and Future Work**

In this paper we focus solely on image classification domains for simultaneous architecture search and transfer...
Our current CENAS implementation relies on a fairly straightforward evolutionary search process. However, given that this search space is unbounded, it is unlikely that we have discovered the true global maxima. Simply increasing the number of generations or the population size is unlikely to solve this problem. We are currently exploring alternative strategies for more fully exploring this space, including ways to estimate the probable value of different operators in certain locations of the space or enforcing diversity with approaches like MAP-Elites (Mouret and Clune 2015).

Conclusions

In this paper we argue for exploring the problem of simultaneous architecture search and transfer learning as it relates to combinational creativity, and introduce an approach we call Conceptual Expansion Neural Architecture Search (CENAS). This approach relies on a neural representation of combinational creativity, the ability of humans to combine existing knowledge to produce novel solutions. We compare our approach to a set of baselines on several experiments using well-known image classification domains. From this, we identify CENAS as a fast and sample-efficient method that produces high-quality and compact models.

Ethics Statement

There are a variety of potential concerns for any approach that seeks to lower resource requirements to apply deep neural networks. Specifically, there are ways in which bad actors could theoretically use an approach like CENAS to, for example, derive an image classifier for a particular person faster and with fewer images of said individual. While we did not explore it in this work, prior work with Conceptual Expansion considered the generative case along with the discriminative case (Guzdial and Riedl 2019). Thus, there is a possibility that one could employ CENAS to more easily produce things like “deep fakes”. However, concerns of this nature are premature, given that right now CENAS has only been evaluated in one domain. To combat this potential in future work, we intend to explore how CENAS models can be identified from their output.

Acknowledgments

We gratefully acknowledge Vellore Institute of Technology -Semester Abroad Program for supporting this research. We also acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC).
Figure 4: Example of three different architectures from the final CENAS generation for the held out airplane class.

References


Levy, O., and Markovitch, S. 2012. Teaching machines to
learn by metaphors. In Twenty-Sixth AAAI Conference on Artificial Intelligence.


Punch Card Knitting Pattern Design in Collaboration with GAN

Virginia Melnyk
PhD candidate
Tongji University
Shanghai, China 7GP4+27
vemelnyk@clemson.edu

Abstract
Punch card knitting codifies the different stitch patterns into binary patterns, informing the knitting machine when to change color or to generate different stitch types. This research explores the collaboration with Generative Adversarial Neural Networks (GAN) to generate new punch card pattern designs. Reflecting on the creative collaborative approach to working with artificial intelligence for design. The hypothesis is that AI can learn the basic underlying structures of the punch cards and the pattern underlying structures that is inherent to Fair Isle knitting. Using a dialectic process of curating data sets, to re-configuring data post processing, AI and human design both play a role in the creation of these new patterns. Utilizing Style GAN2, the results from these explorations offer different insights into pattern design and generate new unique designs from the latent patterns. Ultimately the designs are physically tested on a domestic knitting machine, resulting in novel fabrication methods to produce AI designs as a physical result going beyond just the typical computer-generated image.

Introduction
Graphic patterns are all around us, in nature, in mathematics, and in textiles. These patterns are made of repetitive shapes and geometries. Textiles are associated with patterns, as many designs emerged based on the structure of weaving and knitting (Stewart, 2015). Knitting uses a single yarn looped around itself in rows, and to make patterns, multiple colored yarns can be knit together. Similar to the Jacquard loom for weaving, domestic knitting machines use punch cards as a basic binary pattern telling the machine to knit either color “A” or “B”. Most knitting machines come with a set punch cards, and more punch cards can be purchased separately. Images of these punch cards can also be found on the internet, providing a sufficient source for a data set in this research. See Figure 1.

AI is modeled after the human brain, and it is proven successful at learning and understanding patterns in data sets. This research explores what AI can learn from a data set pertaining to knitting punch card images. The hypothesis for this research is to generate new knitting pattern designs using Style GAN@ trained on images of punch cards as a database. These new patterns will result in a collage and distortion of a variety of styles, cultures, and histories from this input data. The results from the StyleGAN2 training are images. These images are then translated into physical punch cards that could be used to fabricate knitted test samples. The results reflect on what can be learned from the knitting patterns designed with AI, revealing the underlying structures of all the patterns based on the input dataset.

Fig 1. Sample Portion of Punch Card Data Set.

The collaboration between the computer and the human designer is in the curation of the data set and selection of viable results. This has inherent bias, as to what images are available on the internet to use for the data as well as certain taste and styles that may be more likely selected by the designer. The GAN algorithm learns from these images and generates results. The human in response learns from those results and adjusts the inputs and weights in the algorithm, to control the creation of knittable patterns and aesthetically pleasing patterns. The research addresses the importance of the history of textiles in computation and goes beyond the textile community as these results begin to discuss a larger question of design computation, ornamentation, craft, and the creative process in collaboration with AI.
Context

Historically textile design has been largely a female dominated craft (Barber, 1995). Meanwhile the computer industry has recently seemed to be more male dominated (Lee, 2019). Although there has also been a long history of textile artists using computers and the feedback loop between the two modes of working (the Center for Craft, 2020). Such as in the works by Janice Lourie, Sonia Sheridan, Lia Cook and others.

Even early computers are based on textile industry. Joseph-Marie Jacquard in 1800’s invented the Jacquard-loom, allowing for complex weaving patterns to be made using punch cards (Essinger, 2007). In developing early computers that utilize binary code, Charles Babbage and Ada Lovelace adapted the use of punch cards for their Analytical Engine (Essinger, 2015).

This is one way the history of punch cards, computation, and textiles is interrelated. Furthermore, with pixels, these the early 8-bit computer graphic images are similar to the pixel like designs generated with knit stitches in knitting patterns. Using artificial intelligence as a way to design new textile patterns combines these old and new methods of textile creation and computing together in a contemporary way.

Computational Textile Design Precedents

There are precedents of designers and artists who have worked with algorithms and AI for knitting, sewing, and embroidery. These examples show some of the development for computational textile design.

Neural Networks have been used for various textile designs, such as for the color selection and pattern design for new embroidery samplers. In this case, a sentence was input into an Entertainment AI that adapted the sentence’s content into the color selection and motifs for an embroidery sampler design (Smith, 2017).

Another example of knitting being explored with generative AI design is through the development of using neural networks to generate CNC knitting machine patterns. This example generates knitting patterns from images of unknown knit material by being trained on the structure of several sample knits and their corresponding patterns, resulting in a user-friendly interface called img2prog (Kaspar, 2019).

Alternatively, hand knitting patterns are written out in a shorthand language, referred to as knit-speak. In the skyKnit project a natural language learning AI was trained on 500 patterns to develop new knit-speak directions for hand knitting patterns. Using the online community of Ravelry.com these patterns were physically knit by artisans and crafters; the resultant designs were ultimately very strange looking (Shane, 2019).

These examples show the development of computation within textiles and design. Eventually, creating actual physical manifestations of the crafted knit work is essential. Currently, so much of AI designs happens and remains within the computer and this research hopes to use AI for pattern design to bring the design into the physical world with physical constraints.

Knitting Patterns

Patterns are repetitive, symmetric, geometric, and balanced; our human brains are for some reason attracted to them. Gestalt theory attempts to outline some of the principles such as the orders of symmetry, figure-ground, similarity, and common fate as ways to describe how our minds begin to understand patterns as a whole before they recognize the specific elements (Koffka, 2013). Psychologists are still studying the ways that our minds process these patterns.

Since neural networks are modeled after how our brains learn, artificial intelligence predictably should understand the specific rhythms, symmetries, geometries, and spacing that make these knitting patterns. Although Neural Networks are learning these patterns from localized relationships, our brains recognize the overall patterns. This sets up an interesting dichotomy as the task is approached in opposite ways, but the results are potentially the same.

Punch card knitting patterns interoperate a traditional knitting style called Fair Isle. Its origins are credited to Scotland’s Fair Isles, as it is a popular knitting pattern technique in that region. Fair Isle patterns are recognizable by their basic geometric shapes, small-scale repetition, mirroring, and simple color changes that never consist of more than two colors per row. See Figure 2.

Fig. 2. Example Fair Isle Knit Pattern.
While one color is used as activate stitches, the other colored yarn floats in the back. Floats should be no longer, than three to five stitches in a successful Fair Isle pattern (Pulliam, 2004). The switching off and on between colors creates a pattern through pixel-like imagery as each stitch acts like a pixel of color. The use of only two yarns at a time makes this knitting technique ideal for the binary codification into punch cards, although this limits the types and styles of possible patterns that can be generated. Furthermore, Fair Isle knit pattern punch cards have an almost 1:1 relationship with the image of the punch card, this provides an easy starting point to design AI knit patterns as the designer can visually see the potential design in the punch card results before having to test knit the patterns.

**Punch Cards**

Domestic knitting machines were popular between the 1940s until the 1980s. The hobby has since decreased in popularity, resulting in many of the companies that sold knitting machines and punch cards no longer producing them. However, there are many images of punch cards and patterns available online.

A standard punch card is 24 dots or stitches wide and about 60 stitches long. When knitting the patterns can be repeated in the vertical direction and in the horizontal direction to create larger knit fabric pieces.

There are three main types of Fair Isle patterns: geometric patterns, organic or floral patterns, and object-based imagery.

For this project the database of knitting punch card images was generated by image scraping from Google and incorporated all the different styles. These images were sorted manually to affirm the best quality of images for training, culling out images that are un-clear, low resolution, images of knit material rather than punch cards, and images that are not straight on. This process resulted in a concise set of data of black and white legible punch card images. See Figure 1.

This sorting could generate bias, as certain images that were removed may have been interesting patterns but were removed based on image legibility.

In order to generate a more extensive set of data from the small data set of quality images collected, the punch cards were cropped down into smaller sections. As punch cards have a defined width but an undefined length, the various lengths would cause issues in training, thus cropping them into equal sized images would produce more consistent data set. Each image was cropped into square proportioned images, resultantly showing 24 dots by 24 dots of the punch card. These images consisted of overlaps between them. This cropping of the images is a common technique as the GAN training we are interested in the localized relationships rather than the overall pattern design.

Mirroring was also used, as well, since punch cards do not necessarily have a front or back and can be fed into the knitting machine facing either direction. Therefore, some of the asymmetrical punch cards were fed into the data set facing multiple directions.

This achieved a final set of about 1200 punch card images for training. Although, with in this data set, bias may be hidden within the Google Images as the search results from what is available on the internet. Perhaps certain patterns may appear more frequently than others due to cultural popularity and preference. The image searches were also run in English language search terms, and knitting patterns from English websites would come up more frequently than those from other languages. This could favor certain styles of patterns that are more popular with in English speaking cultures. In addition, Fair Isle style is from the United Kingdom, thus many of the patterns designs created have a western sensibility and are represented in the cultural significance of these patterns. Many patterns, such as polka dots to stripes, have certain cultural meanings and historical significance, but they can be different meanings in another culture (Stewart, 2015).

This research does not hope to look at one culture over the other or to remove cultural significance from these patterns. It attempts to perhaps understand and learn the deeper localized relationships of geometry and proportions that generate patterns across cultures and meaning.

Furthermore, to understand the underlying structure of the punch cards, for the specific constraints in Fair Isle knitting, such as not to have long floats. The pattern should also be repeatable; it would need to have balance across the card rather than be weighted to one side or the other. Furthermore, the knit is constructed in rows, and each row can exist independently, but a successful pattern has vertical and horizontal repetition and geometry.

**StyleGAN2**

StyleGAN2 was released in early 2020 by NVIDIA, and it is an update to the earlier StyleGAN developed in 2018. Generative Adversarial Networks (GAN) consist of two neural networks, one which generate images and one that test the images (Karras, 2020). StyleGAN2 learns the characteristic artifacts in a data set of images to produce new images.

The GAN first generates images from a random noise pattern while the discriminator tests them and feeds back in-
formation to the generator to correct it. Each time the data set is processed is an epoch, in which the generator gets closer to the desired results until it eventually the generated images can fool the discriminator into believing that the image is real image. See Figure 3.

In this study the data set was uploaded to a base model of StyleGAN2, pre-trained on bird illustrations. Several tests were run at different ranges of epochs. Through this the designer is able again to curate and collaborate with the GAN to select various weights and adjust the influence of the training. Although StyleGAN2 is a supervised learning, even more control is given to adjusting the settings.

In this project at around 1500 epochs, the images started to begin to look like new punch card designs. At less epochs the designs still consisted of splotches of solid black from the originally trained bird illustrations. Where training longer 1500 epochs, the model began to face mode collapse. The punch cards generated at this point began to look all self-similar, and the individual dot matrix was became lost. This is most mode collapse likely due to the data set being too small and self-similar. Another cause could be failure to converge. Further investigation of this could be explored into what may cause these failures.

![Fig. 4. Image Results from StyleGAN2 Training.](image1)

Ultimately, the results from 1500 epochs was a set of 50 successful sample images. The images appeared to have the basic structures of punch card designs. The designer could judge the success of these images based on their appearance to look like other known punch cards. See Figure 4. Since these images were in the square format, three images close in aesthetic quality were selected and combined vertically to create a punch card pattern in the similar proportions of a typical punch card.

In the resulting images there is a variety of differences between images with a high density of dots to ones that were relatively sparse. Some patterns seemed very random while others had clear underlying diagonals, checkers and other patterns embedded within them. Although these that appeared random at a glance, it did have some underlying structure revealed upon further reading and inspection. Patterns did emerge, such as checkered patterns and diagonal stripes and vertical designs. These types of repeated structures can be seen in many of the input patterns from the data set.

![Fig. 5. Translation of image training to Knit Results.](image2)

**Results**

After the designs were digitally generated, physical punch cards were made. The image results from each of the methods was not clear enough to directly use as a punch card and needed to be processed. Grasshopper for Rhino was used to trace the large, clear dots from the images into vector line work, which was then organized on the grid structure by moving these circles to the closest grid points. See figure 5.

The patterns were then laser cut out of thick Mylar to make them into usable punch cards for the knitting machine. They were then used to knit on a Brother KH836 Domestic Punch Card knitting machine with a standard 4.5mm gauge. Since the punch card pattern is only 24 stitches wide, this would result in a small pattern of only four inches wide. Therefore, the pattern was set up to repeat once in width. This created an eight-inch by eight-inch test swatch of material, allowing it to knit once vertically through the pattern design.

Two different colored yarns were used to visually and texturally make the pattern apparent. Physically knitting the patterns gives a better understanding of the successes and failures of the Fair Isle knitting punch cards, as they could be tested with material constraints of the different yarn types and the physical knitting process.

**Physical Knit Results**

The StyleGAN2 training resulted in a range of outputs, consisting of images with very dense dots to very sparse dots. Some of the sparse dot patterns were potentially going to have an issue, as there were some rows with only one or two changes in color, resulting in very undesirable long floats. Although, these patterns did result in having other features such as clear vertical and repetitive structures. See figure 6.

The denser StyleGAN2 generated patterns were more successful punch cards as they had adequate spacing for short floats, consisting of lengths under six stitches. In addition, because the pattern changed colors so often it is difficult to tell that the pattern is repeated more than once horizontally which is quite successful. This pattern was
more of an overall discrete dot hatch or fill between the two colors rather than an image-based pattern. The resulting pattern ultimately has a certain movement to it as it resonates between the different diagonals and checkered designs. See Figure 7.

Fig. 6. Results StyleGAN2 Pattern example 1

Fig. 7 Results StyleGAN2 Pattern example 2

Patterns Results Compared to Random

Although the patterns resultantly look random at first glance, they represent a binary pattern of 24 by 24 dot matrix the possible number of patterns is $2^{(24\times24)}$. This is an extremely large amount. Subsequently, if a random binary code generator was asked to make an array of 576 numbers there would be no logic as to how these numbers may arrange on the 24 by 24 punch card grid. As well, the results may lead to possibilities where there is only one dot in a full array of 0’s. These types of imbalanced results from a completely random pattern wouldn’t generate successful knitting patterns.

This suggests that the resultant patterns which may look random are far from it. That there are learned qualities about proportion of dots, spacing of dots, as well as not having too many or too little in a row.

Furthermore, these patterns look random at first glance but further study and inspection of both the punch card, and the knits do show that there are clear successful underlying structures learned. They had a noticeable clear structure of diagonal pattern and checkered patterns, disrupted by some random stitches. This is possibly because the data set had a lot of diagonal patterns in it. Additionally, these geometries work well for the constraints of Fair Isle knitting and the knitting machine functions.

It is difficult to fully tell whether these underlying patterns are really existing or whether it is the human brain is just imagining patterns where there are none. The Gestalt theory supports the ideas that our brains are wired to find patterns, structures, and logics in the world around us to help us make sense of our surroundings.

Conclusion

Each of the resulting tests developed unique patterns that never existed before. The results did have success, and there were clear underlying structures that each of the training method could understand and replicate patterns that seemed to fit within the constraints for Fair Isle knitting.

Throughout the process the human and the computer collaborated. First, the designer worked to search and curate the data set. Next, the data set was too small, therefore it needed to be multiplied. This was done by cropping the images into multiple smaller images and mirroring the images, to generate a larger data set for training. Then, the human designer also collaborated when running the StyleGAN2 training specific weights and epochs were tested to get the desired results further supervising this training. Finally, once the images were output from the training they still need the designer to select and reconfigure the images into longer ones to make typical knitting punch cards.

Since the images were also not perfectly clear, Grasshopper and Rhino were used to re-configure and refine these images into usable punch cards. The knitting was then all done by the human craftsman. This process uses a lot of back and forth collaboration between AI and algorithms to design and create these new punch card patterns.
Resultantly, the neural networks learned the patterns underlying structures, which has noticeable features from the styles in the existing dataset. These underlying structures worked to create visual appeal that are essential to the knit material's tectonics as well as the principals of patterns that For the development of this research, there are still opportunities to have more control over the data, such as inputting specific pattern types such as only the geometric patterns, or testing punch cards with tuck or lace patterns rather than Fair Isle.

These patterns also have further potentials as to how they can be utilized in real world situations as fashion, décor, or architecture. The possibilities to combine ornamental patterning with functional aspects of using different materials for texture and material proper changes such as elasticity are the next phases of this design research. These patterns ultimately combined the structures and mish-mashed the cultural significance behind these zig-zags, dots, and diamonds into something new and designed computationally with collaboration between human designers. This serves as a reflection on our historical significance of textiles and computation as well as posing the design potentials of the new age of AI and technology in our daily lives.

References
Karras, T. L. e. a., 2020. Analyzing and Improving the Image Quality of StyleGAN. s.l., Computer Vision Foundation.
Shane, J., 2019. You Look Like a Think and I Love You: How artificial intelligence works and why it's... making the world a weirder place. New York: VORACIOUS Press.
Shape Inference and Grammar Induction for Example-based Procedural Generation

Gillis Hermans, Thomas Winters and Luc De Raedt

Computer Science Department
KU Leuven, Belgium
gillis.hermans@student.kuleuven.be; {thomas.winters, luc.deraedt}@kuleuven.be

Abstract

Designers increasingly rely on procedural generation for automatic generation of content in various industries. These techniques require extensive knowledge of the desired content, and how to actually implement such procedural methods. Algorithms for learning interpretable generative models from example content could alleviate both difficulties. We propose SIGI, a novel method for inferring shapes and inducing a shape grammar from grid-based 3D building examples. This interpretable grammar is well-suited for co-creative design. Applied to Minecraft buildings, we show how the shape grammar can be used to automatically generate new buildings in a similar style.

Introduction

Procedural modeling (Parish and Müller 2001) and procedural content generation (PCG) (Yannakakis and Togelius 2018) are used to co-creatively and automatically generate content for applications such as video games, films and simulations. As the complexity and scope of these applications grows, these methods are increasingly relied upon to generate content. Yet, the creation of procedural rules that can generate a particular style and content is a difficult and time-consuming process (Štava et al. 2010). Instead of creating rules by hand, it is possible to learn rules from examples. Furthermore, one of the critical challenges of PCG is style inference, or the ability to generate content in the same style that has been learned or inferred from examples (Togelius et al. 2013). Solving these challenges could also be considered a step in the direction of computational creativity (Toivonen and Gross 2015), as learning new styles is a crucial part of their further exploration.

In this paper, we propose SIGI (Shape Inference and Grammar Induction), a novel method for inferring the style of one or more grid-based 3D buildings in the form of a shape grammar (Stiny 1980). This grammar of geometric designs defines the style as a set of building style features, such as columns and windows, and their relations, and can be used to generate new buildings in a similar style. We make the following contributions:

• We propose a method for inferring shapes present in grid-based 3D buildings. Unlike most previous work, shape inference allows the segmentation of examples with limited user input and without predefined feature classes.
• We show how these shapes are used to induce a shape grammar, which allows co-creative design and automatic generation of similar buildings in an interpretable way. Furthermore, SIGI allows for the induction of a shape grammar from multiple example buildings, which generate buildings in a shared style. Applied to Minecraft examples, we demonstrate our approach and results.
• As a notion for repetition or symmetry in the examples, we define matching shapes that lead to an enlarged generative space and the generation of novel buildings.

Shape Inference

In this section, we introduce shape grammars and discuss shape inference, a component of SIGI that seeks a set of shapes that correspond to parts of the style present in the example buildings. We further use style feature as an informal notion for any building component, such as a window, wall, balcony or awning, that is part of the building style.

Definitions

Given grid-based 3D buildings $E$, SIGI infers shapes $S$ and induces a shape grammar $G$ for the style of $E$. Input examples $E$ are composed of elementary components at a position in $E$. We use Minecraft buildings as examples, which exist out of a set of elementary blocks $B$. A block $b$ is a tuple $(t,p)$ with a type $t$ and a position $p = (x,y,z)$ in a grid-based 3D coordinate space in $Z$. A block represents a voxel at $p$ in a Minecraft world textured according to its type $t$.

We adapt the shape grammar formalism (Stiny 1980) for grid-based shapes as a 4-tuple $(S, L, R, I)$ where:

• $S$ is a finite set of shapes
• $L$ a finite set of labels
• $R$ a finite set of shape rules of the form $\alpha \to \alpha\beta$ where $\alpha$ and $\beta$ are labeled shapes $(S, L)$
• $I$ the initial labeled shape of the form $(S, L)$

A shape $s$ in $S$ is a set of blocks $B_s$. $B_s$ is a subset of blocks $B$ in an example $e$ in $E$: $B_s \subseteq B$. Blocks $B_s$ are connected such that all blocks in $B_s$ are reachable from every other block in $B_s$ by following a path through adjacent blocks in
Figure 1: Example (a) and inferred shapes: 2D with $\alpha = 0.75$ (see Equation 4) (b), rectangular with $\alpha = 1.0$ (c) and 3D with $\alpha = 0.25$ (d). Inferring with merge operations and overlap. Shapes were rotated and matching shapes were combined for sake of clarity.

Figure 2: Matching vertical (a) and horizontal (b) shapes rotated along the $z$-axis. Two non-matching shapes with the same block configuration for which no $\tau_m$ exists (c).

Inferring Shape Sets

In order to infer a suitable set of shapes $S$ for $E$ we apply a local search that minimizes a cost function on $S$. Aside from ensuring shapes meet their requirements, we strive to find shapes that form a suitable description of the style. Each shape $s$ ideally matches a style feature present in the examples. Shapes are not limited to predefined feature classes, as in other work (Teboul et al. 2013; Martinovic and Van Gool 2013). Instead, we infer a suitable set of shapes that are likely style features.

We make the assumption that most style features consist of a few components in a limited number of materials or block types. A window, for example, usually exists out of glass and a frame in another material. While this assumption does not hold for any style feature, it establishes a foundation for the inference of shapes that are likely style features. Thus we strive to find simple shapes (containing few block types), such that we avoid representing multiple style features in a single shape. At the same time, we limit the total number of shapes, to avoid overly simple shapes devoid of any meaning.

Cost function

A suitable set of shapes consists of shapes that are neither too simple nor too complex. We introduce a cost function that strives to find this balance.

Firstly, we limit the complexity of shapes by increasing
their cost. As a measure for this complexity, we use entropy $E_s$ (Shannon 1951), or the measure of information content, of a shape $s$:

$$E_s = - \sum_{i=0}^{n} P(t_i) \log_2 P(t_i)$$

(3)

where $n$ is the number of block types in $s$ and $P(t_i)$ is the probability of block type $t_i$ in $s$. The entropy cost favors compact and homogeneous shapes (Liu et al. 2011), such as shapes consisting of just a single block. To counterbalance this, we introduce a cost for the number of shapes $\#S$ in the set, adding by one to remove a bias for shape sets of size one. By favoring a smaller number of shapes, the cost promotes larger shapes. The resulting cost function to be minimized is:

$$(1 + \#S)^{\alpha} \sum_{i=0}^{S} E_{s_i}$$

(4)

where $s_i$ is a shape present in the shape set $S$ and $\alpha$ is a parameter weighing the importance of $\#S$.

**Local search** The cost is minimized by repeatedly executing operations on $S$. We use a hill-climbing algorithm that, at every step, evaluates and applies the first operation on the shape set that decreases the cost. The algorithm converges to a local optimum, once no more operations exist that decrease the cost. Figure 1 shows resulting shapes for different shape specifications and $\alpha$ values inferred from a simple example.

We define two operations on shapes. A merge combines $s_i$ and $s_j$ into one $s_n$, resulting in a new set of shapes $S'$:

$$S' = (S \setminus \{s_i, s_j\}) \cup \{s_n = \{s_i \cup s_j\}\}$$

A split splits $s$ into two shapes $s_i$ and $s_j$:

$$S' = (S \setminus \{s\}) \cup \{s_i \subset s, s - s_i\}$$

Not all operations are legal, as resulting shapes must meet the defined requirements. They must form coherent segments of $E$ and adhere to the chosen shape specification.

During the execution of the local search, it is possible to use the merge, split or both operations. These schemes require different initializations of $S$ at the start of the algorithm: minimal when merging, maximal when splitting and any in between for the combination. When merging rectangular or 2D shapes, once two shapes have been combined they can no longer be separated. One of the axes becomes fixed, locking the blocks out of potentially better shapes in other planes. In order to alleviate this issue, we initialize each block as three shapes, with one in each plane, and only allow merges between shapes in the same plane. After hill-climbing, we ensure all blocks in $E$ are present in $S$ and remove redundant shapes that are entirely covered by others. A side effect of this optimization allows overlapping shapes, which can contain the same blocks, as in Figure 1. While the resulting shape set is no longer a pure segmentation of $E$, allowing overlapping shapes may increase the number of matching shapes, as illustrated in Figure 3. Some blocks may indeed belong to multiple style features present in the examples.

**Shape Grammar Induction** We discuss how the inferred shape sets form a shape grammar, which can be used to produce new similar buildings. Figure 4 shows a partial shape grammar induced by SIGI and an example derivation.

**Shape Rules** Given the shapes $S$ for examples $E$, we induce a shape grammar $G$ with a set of shape rules $R$. When a block $b_i \in s_i$ is directly adjacent to a block $b_j \in s_j$, these form two rules (as in Figure 4):

$$s_i \rightarrow s_is_j \text{ and } s_j \rightarrow s_js_i$$

(5)

When the shape on the leftmost side of the rule is present in the production, the second shape can be added to the production. Every shape is labeled with its original position and orientation in the example structure, such that during production of a shape it can be taken into account to calculate its new position. A transformation $\tau_m$, of the same form as $\tau_m$ in Equation 1, can be applied to the initial shape to move the shape to any other position. When deriving a rule, if the leftmost shape has been transformed by $\tau_m$, the same transformation is applied to the rightmost shape. Thus, the relative positions of both shapes in the example are retained in the production. In this form $G$ generates just subsets of $E$. In order to generalize the generative space of $G$ we make use of matching shapes in the shape set.

**Shared rules** While matching shapes represent the same style features in different positions in the examples, they form rules with different shapes. We share rules between matching shapes, such that they can be applied to multiple shapes in $E$. As an intuitive example, when a balcony is present next to a window in $E$, it can be produced next to any matching window shape in the production. The production rules in Equation 5 extended to (as in Figure 4):

$$m_{s_i} \rightarrow m_{s_is_j} \text{ and } m_{s_j} \rightarrow m_{s_js_i}$$

(6)
Figure 4: Partial shape grammar induced from the example and inferred rectangular shapes in Figure 1. This figure shows 4 (of 16) shape rules from Equation 5, the updated shared rules from Equation 6 and an example derivation of these rules. Matching shapes \( s_2, s_3, s_4 \) form shared rules that enlarge the grammar’s generative space.

where \( m_s \) is the set of shapes that match \( s \), including \( s \). These rules can be seen as shorthand for adding a duplicate rule for each shape that matches the leftmost shape in Equation 5. Starting from a shape \( s \) in \( m_s \), it is possible to expand a shared rule to add \( s_j \) to the production. The transformation \( \tau_m \) in Equation 1, that maps \( s_i \) to its matching shape \( s \), is applied to the new production \( s_j \). Thus, \( s_j \) is transformed to form the same relative position with \( s \) as was present in the example with \( s_i \), as shown in the derivation of Figure 4.

The sharing of rules between matching shapes allows rule expansions outside the space of \( E \). Thus, more matching shapes in \( S \) generalize the generative space of \( G \). Additionally, if two buildings in \( E \) have matching shapes, these two examples will be linked in \( G \), because the matching shape rules provide a bridge between both production spaces.

Production of Similar Structures

The induced shape grammar \( G \) allows the production of new artifacts in a similar style as the examples. Starting from a production \( P \), which contains the initial shape \( I \) chosen from \( S \), shape rules are applied that add shapes to \( P \). At every step the production selects a shape \( s \) from \( P \) and a rule \( r \) that applies to \( s \). For all applicable rules the leftmost shape \( s_l \) is either \( s \) or a matching shape of \( s \). The rightmost shape \( s_r \) is added to \( P \) after applying \( \tau_p \), if applied to \( s_l \) and \( \tau_m \), if \( s \) is a matching shape of \( s_l \). These transformations align the relative positions of the shapes in the production. A new shape is chosen from \( P \), and the process can be repeated indefinitely. When used as a co-creative tool, the designer controls the rule derivation and chooses a stopping point, both of which are hard to do sensibly automatically.

When using \( G \) for automatic generation, rules are applied randomly until a predefined stopping condition, such as a maximum number of rule applications, is reached. Shape grammars are not suited for automatic derivation, because unconstrained derivation frequently adds new shapes to the production (Wonka et al. 2003) and leads to unstructured buildings, as in Figure 5(a). While a designer can guarantee structurally and creatively consistent artifacts by choosing which rules to expand, the quality of automatically generated artifacts is much more difficult to ensure. Consequently, the derivations are usually done by hand or co-creatively with the assistance of a computer. The automatic generation of satisfactory artifacts requires additional constraints, which are difficult to define in the shape grammar itself (Merrick et al. 2013).

Enclosure constraint By removing redundant shapes or filling in empty spaces in an unstructured building, we form a coherent enclosed building with a clear distinction between the in- and outside. We define the enclosure constraint as follows. Rectangular and 2D shapes are restricted to a single position on the \( x, y \) or \( z \) axis. As such \( s \) has two distinct sides side_1 and side_2 on either side of this fixed axis. These consist of the positions of every block present in the shape shifted by 1 or \(-1\) along that axis. The shape \( s \) is enclosed when either side_1 or side_2 can not be reached through a path of empty space starting from the exterior of the structure. We do not extend the enclosure constraint to 3D shapes, as there are no obvious sides to these shapes. A simple pathfinding algorithm explores the production space and finds reachable sides in the production. Once all sides have been explored, we remove any shape for which both sides were reachable, providing an enclosed production, as in Figure 6.
Evaluation and Results

We implement SIGI as a filter for MCEdit-Unified, a world editor for Minecraft. Filters are written in Python code to extract and edit information from a Minecraft world. We evaluate the shape inference procedure through experiments on the effects of its parameters and perform a qualitative evaluation on the results of automatic generation.

Shape Inference Evaluation

SIGI provides a number of parameters for shape inference: three shape specifications and search operations, the $\alpha$ parameter and overlapping shapes. The following experiments address the following question:

Q How do the inference parameters affect the results?

As the ground truth shape set of an example is undefined and ultimately comes down to the intentions of the designer, we emphasize objective summary measures instead of the correctness or value of the resulting shapes. We perform shape inference on each example while alternating all parameter combinations and take aggregate measurements on the resulting shape sets. These are: the number of shapes $\#S$, the percentage of matching shapes $\%M$, the number of blocks in a shape $\text{Size}$ and the number of block types over the shape size $C$ as a measure for complexity. These allow us to estimate the average effects of the examined parameters in the composition of the shape sets.

Data The 9 examples used in these experiments were chosen to encompass various complexities and structural features, such as slanted roofs and cylindrical buildings. Examples 1 – 3 were built for basic testing and 4 – 9 were built by community members and edited to remove excess details. Example building sizes range from 73 to 498 blocks and 5 to 16 different block types.

Experiments As some blocks may belong to multiple style features in the examples, allowing overlap may increase the number of matching shapes. As shown in Table 1, overlap increases matching shapes in general, but also significantly increases the number of shapes. When considering the results of the examples side by side it seems overlap has a different effect on each example. While effective for some examples, such as $E1$ in Figure 1, it is detrimental for more complex examples with smaller shapes. Thus, we do not recommend the use of overlap in general.

Table 1: Results for overlap with the merge operation.

<table>
<thead>
<tr>
<th>Examples</th>
<th>Overlap</th>
<th>$#S$</th>
<th>$%M$</th>
<th>$\text{Size}$</th>
<th>$C$</th>
<th>$#S$</th>
<th>$%M$</th>
<th>$\text{Size}$</th>
<th>$C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>No</td>
<td>27.23</td>
<td>27.8%</td>
<td>96.33</td>
<td>0.15</td>
<td>6.0</td>
<td>21.9%</td>
<td>31.2</td>
<td>0.11</td>
</tr>
<tr>
<td>Yes</td>
<td>50.04</td>
<td>31.2%</td>
<td>94.26</td>
<td>0.16</td>
<td>13.0</td>
<td>33.3%</td>
<td>22.68</td>
<td>0.13</td>
<td></td>
</tr>
<tr>
<td>E1</td>
<td>No</td>
<td>7.75</td>
<td>15.6%</td>
<td>31.59</td>
<td>0.18</td>
<td>10.0</td>
<td>0.0%</td>
<td>25.0</td>
<td>0.08</td>
</tr>
<tr>
<td>Yes</td>
<td>6.71</td>
<td>29.8%</td>
<td>33.80</td>
<td>0.13</td>
<td>5.0</td>
<td>40.0%</td>
<td>21.0</td>
<td>0.13</td>
<td></td>
</tr>
<tr>
<td>E5</td>
<td>No</td>
<td>33.05</td>
<td>28.1%</td>
<td>137.14</td>
<td>0.14</td>
<td>26.0</td>
<td>8.8%</td>
<td>11.28</td>
<td>0.15</td>
</tr>
<tr>
<td>Yes</td>
<td>72.53</td>
<td>26.9%</td>
<td>136.91</td>
<td>0.17</td>
<td>61.0</td>
<td>27.0%</td>
<td>9.18</td>
<td>0.16</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Results for $\alpha$ parameter values.

The $\alpha$ parameter weights the number of shapes in the cost function. A higher value promotes a smaller set of shapes and thus larger shapes. An $\alpha$ of 0.0 produces minimal shapes because only entropy is taken into account. Table 2 shows that larger $\alpha$ lead to larger shapes and less matching shapes. An $\alpha$ of around 5 results in maximal shape sets, as the entropy is disregarded. Thus any value in this range, specifically around 1.0, will provide reasonably sized shapes for the shape set.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$#S$</th>
<th>$%M$</th>
<th>$\text{Size}$</th>
<th>$C$</th>
<th>$#S$</th>
<th>$%M$</th>
<th>$\text{Size}$</th>
<th>$C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>64.60</td>
<td>52.0%</td>
<td>23.93</td>
<td>0.199</td>
<td>47</td>
<td>57.4%</td>
<td>6.60</td>
<td>0.168</td>
</tr>
<tr>
<td>0.25</td>
<td>56.85</td>
<td>46.5%</td>
<td>43.82</td>
<td>0.165</td>
<td>36</td>
<td>51.2%</td>
<td>6.90</td>
<td>0.165</td>
</tr>
<tr>
<td>0.5</td>
<td>55.29</td>
<td>41.7%</td>
<td>66.67</td>
<td>0.180</td>
<td>28</td>
<td>42.3%</td>
<td>9.04</td>
<td>0.152</td>
</tr>
<tr>
<td>0.75</td>
<td>53.26</td>
<td>39.3%</td>
<td>68.84</td>
<td>0.177</td>
<td>24</td>
<td>40.0%</td>
<td>11.26</td>
<td>0.137</td>
</tr>
<tr>
<td>1</td>
<td>42.85</td>
<td>33.2%</td>
<td>72.39</td>
<td>0.159</td>
<td>16</td>
<td>33.3%</td>
<td>19.27</td>
<td>0.127</td>
</tr>
<tr>
<td>1.5</td>
<td>41.1</td>
<td>30.4%</td>
<td>78.00</td>
<td>0.157</td>
<td>12</td>
<td>30.0%</td>
<td>22.90</td>
<td>0.124</td>
</tr>
<tr>
<td>2</td>
<td>33.26</td>
<td>23.7%</td>
<td>100.28</td>
<td>0.144</td>
<td>8</td>
<td>22.2%</td>
<td>32.50</td>
<td>0.111</td>
</tr>
<tr>
<td>5</td>
<td>31.04</td>
<td>23.6%</td>
<td>106.23</td>
<td>0.139</td>
<td>6</td>
<td>21.4%</td>
<td>35.00</td>
<td>0.109</td>
</tr>
<tr>
<td>100</td>
<td>31.03</td>
<td>23.6%</td>
<td>106.27</td>
<td>0.139</td>
<td>6</td>
<td>21.4%</td>
<td>35.00</td>
<td>0.109</td>
</tr>
</tbody>
</table>

Figure 6: Two examples (a) with rectangular shapes (b) inferred with merges and $\alpha = 1.0$ and a new building (c) generated with the resulting shape grammar and the enclosure constraint. Shapes in the blue outline (d) are present in both examples and allow the combination of the two styles. Shapes were rotated and duplicate shapes were removed for sake of clarity.
as the merge. Thus, it is more likely for these to converge soon, resulting in large shape sets with small shapes. Conversely, the split operation starts from maximal shapes and converges with larger shapes. The merge and combination produce similar results because both start from the same initialization and follow the same initial path of merges. Occasionally a split operation will occur in the combination, resulting in slightly larger shape sets with smaller shapes. Consequently, the resulting shape set is highly reliant on the initialization of \( S \), because the local search scheme converges quickly in the first local optima.

### Table 3: Results for local search operations.

SIGI allows three shape specifications: rectangular, 2D and 3D. As shown in Table 4, less constrained shapes allow larger shapes with less block types. Even when paired with a low \( \alpha \) (as in Figure 1(d)), 3D shapes are often enormous, encompassing significant subsets of \( E \) and lack matching shapes. Thus, buildings generated from these shapes will lack variation. We recommend a more fine-grained approach in the form of rectangular or 2D shapes.

### Table 4: Results for shape specifications.

In conclusion, the combination of operations with rectangular or 2D shapes without overlap and an \( \alpha \) between 0 and 5 generally result in the most suitable shape sets with fair number of matching shapes.

### Shape Grammar Evaluation

We evaluate the shape grammar by means of a qualitative evaluation on the results of automatic generation, that aims to answer the following questions:

**Q1** Is SIGI able to induce a shape grammar capable of generating new buildings that are similar to the examples?

**Q2** To what extent does SIGI infer the style of the example buildings?

Results\(^3\) were generated from the same examples used in the shape inference evaluation with two additional examples for a shared shape grammar. We generated new buildings using 20 and 50 rule productions with and without enclosure.

While the results of an unconstrained automatic derivation of the shape grammar are unusable artifacts by themselves (Figure 5(a)), enforcing the enclosure constraint can produce similar and suitable buildings (Figure 6 and 5(b)). Moreover, SIGI allows the induction of a shape grammar from multiple examples, and the generation of new buildings in a shared style (Figure 6). Results were found with straightforward shape inference parameters: \( \alpha \) of 1.0, rectangular or 2D shapes and merge or the combination of operations. Although these could be tuned to further improve the results, at least for simple examples limited input is necessary. Despite the success of Q1, our approach has a number of limitations.

### Limitations

SIGI struggles with buildings that include complex style features, such as slanted roofs, because they cannot be effectively represented with rectangular or 2D shapes. Instead, these result in many small shapes and rules that complicate the shape grammar resulting in subpar productions (Figure 5(a)). Using 3D shapes leads to a few large shapes without any matching shapes (Figure 1(d)).

The enclosure constraint is limited in a few ways. Enclosure removes shapes even when unenclosed in the original examples. A potential solution marks these during shape inference as unenclosed shapes which are ignored by the enclosure constraint. At the same time it is possible that the removal of unenclosed shapes reveals new unenclosed shapes, as in Figure 5(b). Thus, enclosure can be run multiple times until no shapes are removed. Furthermore, there is no guarantee that any part of a generated building will be enclosed, resulting in empty generated artifacts. Consequently this generation process is not suited for *on the fly* generation, for example during gameplay.

This shape grammar considers only local relationships, in the form of adjacent shapes. Our shape grammar is thus capable of extending shapes arbitrarily to form new structures in a similar style, which can result in novel buildings such as in Figure 6 and 5(b). However, the style of buildings consists of a global structure (Mitra et al. 2014) in addition to the local structure, which SIGI does not take into account. Thus as an answer to Q2, while SIGI is able to capture local style features and how they neighbor each other, the inferred style may be much more general than what we perceive as the style of a building due to neglecting the global structure.

### Discussion, Related and Future Work

As an answer for the challenges of style inference (Togelius et al. 2013) and learning rules from examples (Štava et al. 2010), we proposed SIGI, an approach towards shape inference and grammar induction from grid-based 3D buildings. In this section we discuss SIGI, compare it to related example-based procedural methods and discuss future work.

### SIGI and Related Work

#### Shape Inference

SIGI employs a local search through candidate shape sets, minimizing a cost function, to infer shapes from examples. This cost favors shapes that are likely style features, with the assumption that features contain a few block types. Thus, shapes are not guaranteed to accurately represent style features, especially complex features for which this assumption does not hold. In spite of this, the resulting grammar does not strictly require shapes that correspond tightly to style features. Although these could im-
prove the interpretability of the shape grammar, resulting in a more understandable derivation and modification process. While resulting shapes are often satisfactory for simple examples, the search can get stuck in bad local optima. Adding common local search refinements, such as backtracking and restarts (Aarts and Lenstra 2003) can alleviate this issue. In SIGI each inferred set of shapes is limited to one shape specification. However, style features are best represented by different types of shapes: rectangular shapes are ideal for walls but not for slanted roofs. Thus, combining specifications (not exclusive to the ones defined in this paper) could better represent the examples.

Existing methods that induce shape grammars from 2D building facades rely on predefined feature classes (Teboul et al. 2013) and labeled input (Martinovic and Van Gool 2013). One approach that induces a grammar for 3D buildings (Aliaga, Rosen, and Bekins 2007) requires the user to subdivide the building into basic building blocks by hand. However, model synthesis (Merrell and Manocha 2011) and inverse procedural modeling (IPM) (Bokeloh, Wand, and Seidel 2010), require limited user input in the form of a few parameters or constraints. SIGI requires a similar amount of user input in the form of parameter choices, without any labeled data or predefined feature classes.

**Shape Grammar Induction**  SIGI depends on matching shapes to enlarge the induced grammar’s generative space and generate novel buildings. Even when present in the examples, matching shapes must be found during shape inference. A potential improvement redefines the cost function to encourage finding matching shapes. Resulting shape grammars can be interpreted in a visual manner, as in Figure 4. Shape grammars are ideal for use in an interactive editor, similar to previously defined shape grammar interfaces (Müller et al. 2006; Bokeloh, Wand, and Seidel 2010). Such an editor allows co-creative design of new buildings by guiding the derivation and modifying the buildings and grammar. The shape grammar is not inherently suited for automatic generation, because random application of rules adds new shapes to the production, without concern for the global structure (Wonka et al. 2003). Nonetheless, with the addition of enclosure, SIGI is capable of automatically generating suitable buildings for one or more simple examples.

Machine learning PCG methods, trained on example content, implicitly address the challenges of style inference and automatic rule learning (Summerville et al. 2018). However, two common issues not present in SIGI are the lack of sufficient training data and the uninterpretable nature of many ML approaches. Both model synthesis (Merrell and Manocha 2011) and IPM (Bokeloh, Wand, and Seidel 2010), the most similar approaches to SIGI, process 3D examples with limited user input and generate new structures in a similar style. Just as SIGI, model synthesis allows the use of multiple examples towards a shared style. Finding repetition or symmetry in the examples is an inherent problem in example-based PCG, which SIGI tackles with matching shapes and shared rules. While model synthesis directly synthesises a new model from the examples by finding symmet-
spatial distribution of the style in an orderly fashion, for example disallowing floating doors as present in Figure 6.

While further details must be discussed and implemented in future work, SIGI provides a basis for a 3D example-based split grammar approach.

**Conclusion**

A current challenge for procedural generation is the induction of rules for the style of example content (Štáva et al. 2010; Togelius et al. 2013). We proposed SIGI, an example-based procedural approach towards the generation of grid-based 3D buildings. SIGI induces a shape grammar from inferred shapes that allows co-creative design of similar buildings in an interpretable way. Furthermore, with the addition of the enclosure constraint, it is capable of automatically generating suitable buildings, with a few limitations. A crucial aspect of this challenge is finding repetition and symmetry in the examples that lead to generalizations in the induced generative space, which we tackle with matching shapes and shared rules. Existing methods, including SIGI, do not take into account the global structure of buildings, although it can be considered a crucial part of their style. This paper serves as a stepping stone towards the currently unexplored problem of inducing split grammars from 3D examples, an approach that does take global structure into account and is inherently suited for automatic generation.

**Acknowledgments**

This research was partially funded by KU Leuven Research Fund (C14/18/062) and the Research Foundation - Flanders (GO97720N). Thomas Winters is a fellow of the Research Foundation Flanders (FWO-Vlaanderen).

**References**


Syllable Neural Language Models for English Poem Generation

Danielle Lewis
Department of Electrical and Electronic Engineering
City, University of London
London, UK EC1V 0HB
Danielle.Lewis@city.ac.uk

Andrea Zugarini
DINFO
University of Florence
Via di S. Marta, 3 - 50139 Florence, Italy
andrea.zugarini@unifi.it

Eduardo Alonso
Department of Electrical and Electronic Engineering
City, University of London
London, UK EC1V 0HB
E.Alonso@city.ac.uk

Abstract
Automatic Poem Generation is an ambitious Natural Language Generation (NLG) problem. Models have to replicate a poem’s structure, rhyme and meter, while producing creative and emotional verses. The lack of abundant poetic corpora, especially for archaic poetry, is a serious limitation for the development of strong poem generators. In this paper, we propose a syllable neural language model for the English language, focusing on the generation of verses with the style of a target author: William Wordsworth. To alleviate the problem of limited available data, we exploit transfer learning. Furthermore, we bias the generation of verses according to a combination of different scoring functions based on meter, style and grammar in order to select lines more compliant with the author’s characteristics. The results of both quantitative and human evaluations show the effectiveness of our approach. In particular, human judges struggle to recognize real verses from the generated ones.

Introduction
Automatic poetry generation is an evolving area at the crossroads of computational creativity and NLG (Gatt and Krahmer 2018). NLG is a well-established sub-field of Artificial Intelligence (AI), with the goal of “generating understandable texts in a human language based on non-linguistic communication” (Ehud and Robert 2000). NLG is a challenging problem which has seen huge advances and contributions to Natural Language Processing (NLP) over the last 20 years: producing various types of texts, from biographies (Kim et al. 2002) to weather and financial forecasts (Reiter et al. 2005; Plachouras et al. 2016), as well as creative texts such as story narratives (Gervás et al. 2006), jokes (Ritchie et al. 2007) and poetry (Zugarini, Melacci, and Maggini 2019). Automatic poetry generation is an incredibly interesting topic for artificial intelligence and especially challenging due to the complex language features involved; like syntax, semantics, phonetics and lexical choice (Oliveira 2009).

Poem generation requires consideration to both content and form. Not only is poetry an expression of language, but an expression of the artist themselves. Automatically generated poetry must capture the linguistic features that can characterize a poet; the rational and semantic qualities of the poet’s works, naturally influenced by their personal experiences, beliefs, and literary background.

Modern techniques for poetry generation have largely utilized neural architectures with a post-processing stage to generate well-formed verses. More often this research has used the poetical works of several authors, as opposed to a target poet, to tackle the need for large quantities of data.

(Zugarini, Melacci, and Maggini 2019) proposed a simple neural network model to generate verse in the Italian language, explicitly from the poet Dante Alighieri. What distinguished this approach from previous language models was the use of syllables as input tokens. This intuition for using sub-word information was based on the dependency of poetry using syllables to regulate form; meter and rhyme. The syllable-based approach proved successful for the Italian language, which has a rich morphology. However, it is unknown if the technique could be applied to other languages, specifically to a less phonetic language like English.

In this paper, we extend the syllable-based language model proposed in (Zugarini, Melacci, and Maggini 2019) to the case of English language for a target author, namely the romantic poet William Wordsworth. The model consists of a Recurrent Neural Network that outputs one syllable at each time step, conditioned to the previously generated text. The model is trained using William Wordsworth’s work: The Prelude, composed in blank verse, i.e. unrhymed lines of iambic pentameters. By virtue of the syllable-based approach, the proposed model can learn several properties of the input and has large flexibility in its potential generation. To account for this, generations which resemble The Prelude and Wordsworth’s style are favoured. Neural networks trained on a single author can lead to low generalization due to small training data. To combat this, a multi-transfer learning system is proposed. A transfer of information is learnt by the language model, using Wordsworth’s non-poetic prose, a selection of his production of poems and lastly, the autobiographical epic, The Prelude.

Experimentation demonstrates that exploiting Wordsworth’s production improves the perplexity of the language model, suggesting that the model’s ability to capture the language and contents of The Prelude is enhanced. A qualitative analysis of the generated verses using human evaluation in a Turing style test was carried out. It was found that the generated verses were considered to be real by the judges, even more frequently than the genuine Wordsworth verses. To encourage further
research and the reproduction of the syllable poetry generation model, the open-source code can be downloaded at https://gitlab.com/danielle.evalewis1/ poetrygeneration_william_syl-worth.

The paper is structured as follows. The next Section gives an overview of advances and state-of-art approaches to poetry generation. Then, we describe the proposed model and the generation mechanism, we report the results of the experiments, and finally we draw the conclusions.

Related work

According to the literature, the problem of poetry generation has been tackled often using machines which are programmed to generate poetry or approaches which utilize machine learning. Earlier methods relied on rule-based solutions, while more recent state-of-art techniques have employed learnable language models to tackle flaws of previous systems and go beyond template filling. Language Modelling predicts which word comes next, given a sequence of previous words. Neural language models have been the dominant class of algorithms applied to NLG in the last few years. Neural networks learn representations at increasing levels of abstraction through backpropagation (LeCun, Bengio, and Hinton 2015; Goodfellow et al. 2016). These representations are dense, low-dimensional and distributed, which complements the task of natural language processing by capturing grammatical and semantic generalizations (Gatt and Krahmer 2018). Whilst a feed-forward neural network has been found to be successful to address language modelling (Bengio et al. 2003), recurrent neural networks (RNN) are the much-preferred approach (Mikolov et al. 2010).

RNNs were designed to improve sequence modelling and retain information from sequences of text by introducing memory loops within the network. (Hochreiter and Schmidhuber 1997) developed a more sophisticated RNN architecture called the LSTM (Long Short-Term Memory), designed to retain information for an extended number of timesteps and as a solution to the vanishing gradient problem. Compared to other language models, LSTMs can handle various sequence lengths and avoid data sparseness as well as the explosion of the number of parameters. Several researchers have used LSTMs to produce state-of-art generation systems to generate sonnets from topics (Gharavinejad et al. 2016), automatic rap lyrics (Potash, Romanov, and Rumshisky 2015), and target author-stylized poetry (Tikhonov and Yamschikov 2018; Zugarini, Melacci, and Maggini 2019)).

There are many different approaches to poetry generation and the use of language models. (Lau et al. 2018) used a joint architecture consisting of a word-level language model with both word and character representations, where generations of quatrains are selected based on a pre-processing step. Another combinatory architecture using a language model and topic modelling was proposed by (Van de Cruys 2020) in which the system was exclusively trained on non-poetic text, with a post-processing step to constrain poetic verse. Word-based language models usually involve large vocabularies, storing all the most frequent words in a large textual corpus. They also cannot generalize to never-seen-before words. To overcome these issues, some approaches have exploited sub-word information: (Hwang and Sung 2017) proposed a character-level solution and (Miyamoto and Cho 2016) combined word embeddings with character-level representations. Exploiting sub-word knowledge is crucial to regulate and capture the poem’s form. It has been shown in (Marra et al. 2018) that character-based models can produce powerful word and context representations, capturing both morphology and semantics. However, all of these solutions learned language models from large corpora, based on the works of multiple authors or texts. (Zugarini, Melacci, and Maggini 2019) took the intuition of sub-word information and chose syllables to tokenize text, and trained a syllable-based language model from a single Italian author. Syllables are naturally well suited for poetry, since they govern several aspects of the poem’s form. As the poetical production of a single target author is commonly insufficient to train a deep neural network, they proposed a multi-stage transfer learning solution with other artist’s production and a publicly available modern Italian corpora to capture syntax and grammar.

We follow this approach and extend the syllable-based language model to the English language. Whilst there are partial sets of English syllabification rules, there is no definitive set of rules to follow. Automatically splitting words into syllables is a challenging task, especially because the syllable is difficult to define. Even so, most people agree they can count how many syllables there are in each word or sentence. (Marchand, Adsett, and Damper 2007) state there is a general consensus that a syllable comprises of a ‘nucleus’ which is nearly always a vowel combined with zero or more preceding consonants (known as the onset) and zero or more proceeding consonants (known as the coda). However, for multisyllabic words it is difficult to define which consonants belong to which syllable. A modern alternative to English syllabification is a data-driven or corpus-based approach which tries to deduce syllabifications from previous syllabified words, using a dictionary or lexicon (Marchand, Adsett, and Damper 2007). Since (Liang 1983) formulated his TEX hyphenation algorithm, it has been a standard in the field (Adsett and Marchand 2009).

We show in the Experiments that a syllable solution can indeed be applied to the English language by generating verses with correct form, and share characteristics in the style of the target author.

The Model

The proposed model consists of two blocks: a hyphenation module and a syllable-based Language Model, which processes an input sequence of syllables.

Hyphenation Module

The hyphenation module is responsible for splitting the text into a sequence of syllables. This module is language dependent, because each language has its own rules. As discussed earlier, as opposed to Italian, English does not have a precise set of hyphenation rules. Therefore, we were unable to implement a module similar to (Zugarini, Melacci,
and Maggini 2019), and we relied instead on an implementation of (Liang 1983) algorithm using the python package hyphenate¹, that exploits a TeX approach for finding legitimate hyphenation points. Each verse in the text is converted into a sequence of syllable tokens $x := x_1, \ldots, x_T$ which belong to the syllable dictionary $V_{sy}$. A word-separator is inserted between words in each sequence, to distinguish breaks between words $<$sep$>$, begin-of-verse $<$go$>$ and end-of-verse $<$eov$>$.

**Language Model**

The syllable-based language model learns to estimate the conditional probability of a token given the previous tokens. At each time step $t$, it outputs the token in the vocabulary $V_{sy}$ with highest probability:

$$\hat{y}_t = \arg \max \{ p(y | x_1, \ldots, x_t-1) \}$$

(1)

where $\theta$ are the network’s weights and $\hat{y}_t$ indicates the syllable associated with highest probability. Each element of $V_{sy}$ is encoded into a one-hot representation of size $|V_{sy}|$. The model learns a latent dense $d$-dimensional representation of each token, called syllable embedding. The sequence of syllable embeddings is provided as input to the RNN, collected row-wise in the embedding matrix, one element at each time step. As $V_{sy}$ is the set of all syllables and special tokens, its cardinality is smaller than traditional word-based vocabularies, which means the embedding matrix has significantly less trainable parameters than word-level representations. The internal state of the RNN at time step $t$ is indicated with $h_t$, and computed as follows:

$$h_t = r(e_t, h_{t-1}),$$

(2)

computed by updating the previous state $h_{t-1}$ combining it with the current syllable embedding $e_t$ through the recurrent cell $r$. In our language model, $r$ is an LSTM cell. The hidden state is further projected with a non-linear layer, into a $d$-dimensional vector $z_t$:

$$z_t = \sigma(W h_t + b),$$

(3)

and finally a dense layer back-projects $z$ into the syllables vocabulary space ($\mathbb{R}^{|V_{sy}|}$), that followed by the softmax activation function yield the probability distribution of Equation 1:

$$o_t = W_s z_t,$$

(4)

$$\hat{y}_t = \text{softmax}(o_t).$$

(5)

The language model is trained by minimizing the cross-entropy loss function between $\hat{y}_t$ and the ground truth, i.e. the actual tokens retrieved from Wordsworth’s poetry. It encourages the model to assign high probability to the observed data, pushing toward 1 the element of $\hat{y}_t$ associated to the $t$-th syllable of the current line in observed data. Figure 1 visualizes the structure of the language model through an example.

¹https://pypi.org/project/hyphenate/

**Multi-stage Transfer Learning**

Neural language models are usually trained on large textual corpora. When focusing on a single author’s work, such as in the case of The Prelude: Growth of a Poet’s Mind of William Wordsworth, a language model struggles to learn, leading to poor generalization capabilities. To alleviate the lack of resources, we adopt a multi-stage transfer learning technique to pre-train the model on additional data and then fine-tuning it on The Prelude. We choose this transfer learning approach to mimic the approach used in (Zugari, Melacci, and Maggini 2019), in generating poetic text in the style of Dante’s The Divine Comedy. Highlighting the ability to produce desirable results from a relatively small corpus of a single author’s work by exploiting syllables. In particular, we consider a large selection of Wordsworth’s poetry production (excluding The Prelude) and The Guide through the District of the Lakes in the North of England, a book written in prose. For simplicity, in the rest of the paper the 3 corpora are referred to as The Prelude, Production and The Guide, respectively. We choose Wordsworth’s non-poetic text (The Guide) to grasp the syntax and grammar of the English language as well as the author’s style. The 30 poems from Wordsworth’s production were selected from a bibliography.

**Poem Generation Mechanism**

Once the language model has been trained, we can exploit it at inference time to generate verses.

**Decoding.** After training, new verses can be generated directly from the model. We start with $h_0$ set to zeros, and we feed the system with the start symbol, then we auto-regressively feed the network by sampling the next token at each step. Sampling has proven to be essential for the generation of diverse, creative and free generations (Holtzman et al. 2019). There are several different sampling strategies. We explored two popular sampling techniques: multinomial sampling with temperature (Ackley, Hinton, and Sejnowski 1985; Ficler and Goldberg 2017; Fan, Lewis, and Dauphin 2018) and top-$p$ (Holtzman et al. 2019) sampling. Sampling with temperature regulates the crispness of the probability distribution $p$ through a parameter $t$, namely temperature:

$$p(x_i | x_{<i}) = \frac{\text{exp}(h_i/t)}{\sum_{j=1}^{|V_{sy}|} \text{exp}(h_j/t)}.$$  

(6)

Setting $t \in [0,1]$ skews the distribution towards high probability events, which implicitly lowers the mass of the tail distribution. Top-$p$ sampling, also known as Nucleus Sampling, was instead proposed in (Holtzman et al. 2019). Such stochastic decoding technique achieved higher quality text from neural language models than greedy search and temperature sampling. The approach avoids sampling from the tail of the probability distribution by truncating it dynamically, such that the remaining tokens contain most of the probability mass, at least more than a value $P$. Formally:
Figure 1: Sketch of the syllable-based Language Model. The hyphenation module is responsible for the tokenization of text in syllables (enriched by some special tokens that account for word separation, end of verse etc.). Green rectangles represent syllables’ embeddings, yellow ones the LSTM cell unfolded through time, and the light blue blocks the computations of equations 3 and 4.

\[
p'(x_i | x_{<i}) = \begin{cases} 
    \frac{p(x_i | x_{<i})}{\sum_{x_j \in V^{(p)}_s} p(x_j | x_{<i})}, & \text{if } x_i \in V^{(p)}_s \\
    0, & \text{otherwise}
\end{cases}
\]

where \( V^{(p)}_s \) is the set of tokens constituting the nucleus that as mass probability greater or equal to \( P \).

We keep sampling and generating tokens until the <eov> symbol is generated, or the number of syllables reaches a fixed maximum limit. Numerous different sequences can be generated by sampling from the model’s distribution learned from the training data.

Poem Selection. We generate 100 verses for each sampling strategy and we assign a score \( S(x) \) to each generated verse. \( S \) is an averaging of three different scoring functions: namely \( S_1(x) \), \( S_2(x) \), \( S_3(x) \), based on meter, style and grammar, respectively. To promote verses with an iambic pentameter meter, \( S_1 \) counts the number of syllables (excluding special tokens) in a verse, as follows:

\[
S_1(x) = \text{abs}\left(\frac{|x|}{10} - 1\right),
\]

where \(|x|\) indicates the number of syllables in the verse \( x \) and \( \text{abs}(\cdot) \) is the absolute value function. In this way, verses differing from the 10 syllable target are penalized. To monitor the generation of words in the author’s style, we considered the subset \( V_y^{(k)} \) of top-\( k \) (\( k = 2000 \)) most frequent words (stop words excluded) used by the author in The Prelude. In the attempt to better mimic the artist’s style, we measured with \( S_2 \) the proportion of tokens in the verse that belongs to \( V_y^{(k)} \):

\[
S_2(x) = \text{abs}\left(\frac{|x|}{V_y^{(k)}} - 1\right),
\]

where \(|\cdot|\) is the counting function.

Score \( S_3 \) was used to account for grammar, in the hope of highlighting non-sensical words and repetitions within the line. Python’s LanguageTool\(^3\) was used to count the number of “mistakes” which appear per line, with all generations receiving at least 1 count. LanguageTool is a popular open source proofreading software developed by (Naber and others 2003) checking for grammar, style and spelling. The scores were normalized, resulting in values between 0 and 1 – with 0 being the ideal score.

\[
S_3(x) = \frac{1}{|x|} \sum_{x_i \in x} f(x_i),
\]

\[
f(x_i) = \begin{cases} 
    0, & \text{if } x_i \in V \\
    1, & \text{otherwise}
\end{cases}
\]

where \( V \) is the vocabulary of words accepted by LanguageTool. For all three scoring functions, better performance is demonstrated with a value closer to zero. An average of the three scores was calculated for the 100 generations. The verses from the top 20 highest scores were selected for the human evaluation.

\(^3\)https://pypi.org/project/language-check/
Table 1: Number of verses/sentences and syllables for each corpus.

<table>
<thead>
<tr>
<th></th>
<th>Train</th>
<th>Test</th>
<th>Text style</th>
<th>Syllables</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Prelude</td>
<td>7,134</td>
<td>793</td>
<td>poetry</td>
<td>5083</td>
</tr>
<tr>
<td>Production</td>
<td>17,469</td>
<td>1,941</td>
<td>poetry</td>
<td>6990</td>
</tr>
<tr>
<td>The Guide</td>
<td>948</td>
<td>106</td>
<td>prose</td>
<td>3958</td>
</tr>
</tbody>
</table>

Table 2: Validation and Test PPL for multi-transfer learning. * indicates that the Prelude was fine-tuned using 0.0002 as learning rate.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Val PPL</th>
<th>Test PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prelude</td>
<td>25.36</td>
<td>26.27</td>
</tr>
<tr>
<td>Production→Prelude*</td>
<td>11.92</td>
<td>11.80</td>
</tr>
<tr>
<td>Guide→Production→Prelude*</td>
<td>17.04</td>
<td>18.09</td>
</tr>
</tbody>
</table>

Table 3: Percentage of participants who judged the generations to be real.

<table>
<thead>
<tr>
<th>Author</th>
<th>Real-Mark</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM</td>
<td>56%</td>
</tr>
<tr>
<td>Poet</td>
<td>52%</td>
</tr>
</tbody>
</table>

Experiments

In the following section we report our quantitative analysis of the syllable language model generations using scoring functions, showing also the benefits of transfer learning. Further, we outline the results of the model’s generations using a human evaluation in a Turing like test.

Datasets. The syllable-based language model was trained with three different corpora: The Prelude, Production and The Guide. Corpora statistics are outlined in Table 1. The number of syllables are large in comparison to (Zugarini, Melacci, and Maggini 2019) which highlights the differences in the English and Italian languages. Perplexity (PPL) was measured to choose the best hyper-parameters for the neural network from numerous configurations using validation and test sets from Production. The Hyper-parameter tuning explored different learning rates (with and without decay), batch size, gradient clipping, dropout probabilities and different network sizes.

Training Details. The best performing size $d$ for the syllable embeddings was 300 and the size of the LSTM state was 1,024. Neurons were dropped out with probability 0.3 and the gradient was clipped at norm equal to 4. The size of $V_{sy}$ was set to 8,000, including all the syllables in the three datasets and the special tokens. Regarding the learning, the best results were obtained with batch size 32 and learning rate 0.002. The best parameters were used for pre-training on the Guide and Production and then fine-tuned on The Prelude. Learning rate was tuned to 0.0002 to achieve the best perplexity scores on The Prelude. After a qualitative examination of the best generated verses (i.e. the ones having the highest score S) with both multinomial temperature sampling and top-p using different $t$ and $p$ parameters, we chose multinomial temperature sampling ($t=0.7$) to produce the verses for the human evaluation.

Transfer Learning Results. Table 2 reports the PPL results for validation and test sets from the transfer learning procedure for each dataset.

As anticipated, the language model trained on The Prelude benefits from pre-training on additional datasets. The most significant improvement in PPL is given when pre-training on Production, showing a positive transfer of information from Wordsworth’s poetry production, reducing perplexity by $\sim 53\%$ and $\sim 54\%$ relative improvement in validation and test sets, respectively. The additional pre-training stage on The Guide, was instead not beneficial, probably because of its limited size.

Datasets Val PPL Test PPL

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Val PPL</th>
<th>Test PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prelude</td>
<td>25.36</td>
<td>26.27</td>
</tr>
<tr>
<td>Production→Prelude*</td>
<td>11.92</td>
<td>11.80</td>
</tr>
<tr>
<td>Guide→Production→Prelude*</td>
<td>17.04</td>
<td>18.09</td>
</tr>
</tbody>
</table>

Human Evaluation. The standard evaluation metric for automatic poetry generation uses human evaluation. Human judgement was enrolled to further assess the generations. Judges were recruited in a Turing style test to judge the language model’s generations compared to the author’s real examples. We recruited 15 graduate students, with a mixed background. We refer to them as “non-expert” judges, since they were not specialized in Wordsworth’s production, but had heard of William Wordsworth. They were asked to judge if a given verse was authored by William Wordsworth or not (i.e., generated by our language model). Each judge evaluated 10 verses, 5 of which were from Wordsworth and 5 from our model, as shown in Table 4. The example verses from Wordsworth were from The Prelude. The poem was split into verses, with 5 verses randomly chosen by an algorithm. The 5 verses representing the syllable language model were manually chosen from the top 20 scoring verses from the combined scoring functions. The same ten verses were shown to all evaluators.

Table 3 reports the number of times (percentages) that verses were judged to be authored by Wordsworth. Generated verses from the language model are considered as real 56% of the time, more so than the real examples authored by Wordsworth, with a relative difference of 7.4%. These results match the work from (Van de Cruys 2020) where about half of the generated poems were judged to be written by a human and (Zugarini, Melacci, and Maggini 2019) who achieve a similar 56.25.

Table 3: Percentage of participants who judged the generations to be real.

<table>
<thead>
<tr>
<th>Author</th>
<th>Real-Mark</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM</td>
<td>56%</td>
</tr>
<tr>
<td>Poet</td>
<td>52%</td>
</tr>
</tbody>
</table>

Conclusion

In this paper, we extended the syllable-based approach proposed in (Zugarini, Melacci, and Maggini 2019) to the English Language. We focused on the generation of verses written in blank verse with the style of the poet William Wordsworth. Regardless of differences between the phonetic Italian language and English, the results show the method can be generalized to English, thus proving its potential applicability to other languages, even those with loose hyphenation rules. The adoption of a transfer-learning approach was crucial for alleviating the lack of textual resources necessary to train the neural language model to learn
the poetry of one author. Despite its simplicity and the absence of large-scale collections of data from the target author, our model produced verses that were marked as “real” by human judges over 56% of the time. Apart from transfer learning, such performances were achieved thanks to the poem selection mechanism, which evaluated the generations for meter, style and grammar, and also due to a multi-transfer learning procedure which improves the quality of the model, exploiting a large collection of the poet’s production and prose.

However, the generations produced were small in length and the quality of the generations were not evaluated further for emotion and content. Future work would plan to increase the size of generations and engage expert judges to compare results based on emotion and content qualities, beyond a Turing-style test.

References


<table>
<thead>
<tr>
<th>Example Source</th>
<th>Real Mark %</th>
</tr>
</thead>
<tbody>
<tr>
<td>and when the shadow of the gentler sleep</td>
<td>80</td>
</tr>
<tr>
<td>the intellect of men and hope was theirs</td>
<td>73</td>
</tr>
<tr>
<td>endowed by nature in the midst of airs</td>
<td>60</td>
</tr>
<tr>
<td>beneath the mountain clouds of our two cheeks</td>
<td>47</td>
</tr>
<tr>
<td>for thy own life the errors of the first</td>
<td>20</td>
</tr>
<tr>
<td>and sallying forth we journeyed side by side</td>
<td>80</td>
</tr>
<tr>
<td>if mid indifference and apathy</td>
<td>60</td>
</tr>
<tr>
<td>and from his work this moment had been stolen</td>
<td>53</td>
</tr>
<tr>
<td>and in our dawn of being constitute</td>
<td>40</td>
</tr>
<tr>
<td>even files of strangers merely seen but once</td>
<td>27</td>
</tr>
</tbody>
</table>

Table 4: Examples of verses submitted to judges, some real (Poet), some generated by our model (LM). We also report the percentage of participants who marked each verse as “real”. Marks of the best and worse generated lines are highlighted in bold.
language model. In Eleventh annual conference of the international speech communication association.


Automating Generative Deep Learning for Artistic Purposes: Challenges and Opportunities

Sebastian Berns¹, Terence Broad²,³, Christian Guckelsberger⁴,⁵,¹ and Simon Colton¹,⁶

¹ School of Electronic Engineering and Computer Science, Queen Mary University of London, UK
² Department of Computing, Goldsmiths, University of London, UK
³ Creative Computing Institute, University of the Arts London, UK
⁴ Department of Computer Science, Aalto University, Espoo, Finland
⁵ Finnish Center for Artificial Intelligence
⁶ SensiLab, Faculty of IT, Monash University, Melbourne, Australia

Abstract

We present a framework for automating generative deep learning with a specific focus on artistic applications. The framework provides opportunities to hand over creative responsibilities to a generative system as targets for automation. For the definition of targets, we adopt core concepts from automated machine learning and an analysis of generative deep learning pipelines, both in standard and artistic settings. To motivate the framework, we argue that automation aligns well with the goal of increasing the creative responsibility of a generative system, a central theme in computational creativity research. We understand automation as the challenge of granting a generative system more creative autonomy, by framing the interaction between the user and the system as a co-creative process. The development of the framework is informed by our analysis of the relationship between automation and creative autonomy. An illustrative example shows how the framework can give inspiration and guidance in the process of handing over creative responsibility.

Introduction

The increasing demand in industry and academia for off-the-shelf machine learning (ML) methods has generated a high interest in automating the many tasks involved in the development and deployment of ML models. Such automated machine learning (AutoML) can make ML more widely accessible to non-experts, and decrease the workload in establishing ML pipelines, amongst other benefits.

AutoML is a very active area of research. The progress to date has been documented in several surveys (e.g. Truong et al., 2019; Tuggener et al., 2019; Chauhan et al., 2020; He, Zhao, and Chu, 2021). There exist a book (Hutter, Kotthoff, and Vanschoren, 2019), an AutoML challenge (Guyon et al., 2019) and a dedicated workshop at the International Conference on Machine Learning, currently in its seventh edition. Crucially though, the automation of generative deep learning (DL) as ML subdomain has received very little attention.

While AutoML is concerned with automating solutions for classification and regression, methods in generative DL deal with the task of distribution fitting, i.e. matching a model’s probability distribution to the (unknown) distribution of the data. Neural architecture search (NAS), an important topic of research in AutoML, has been extended to generative adversarial networks (GANs) (Gong et al., 2019; Li et al., 2020; Gao et al., 2020; Fu et al., 2020), one prominent type of generative models. Moreover, evolutionary approaches have been applied to optimising the GAN training objective (Wang et al., 2019) and other training parameters (Costa et al., 2019). Even though certain aspects of the GAN training scheme have been automated, we highlight three gaps in existing research: (i) there exists no unified automation framework for generative DL more generally; (ii) existing work does not address the use of generative DL for artistic purposes; (iii) researchers have not sought to motivate the automation of DL systems with the goal to endow artificial systems with creative autonomy.

We propose a framework for the automation of generative deep learning that, on the one hand, adopts core concepts from AutoML, and on the other hand, is informed by the theory and practice of computational creativity (CC) research, 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, emphasis added). We can leverage insights from CC because automation in generative DL aligns with one of the field’s central research goals: to endow computational systems with creative responsibilities (Colton, 2009), i.e. the ability to make specific decisions in a creative process. These decisions independently can be understood as targets for automation when framing the design of a generative DL pipeline as a form of co-creativity (Kantosalo et al., 2014). By virtue of this interpretation, we can inform the automation of generative DL more specifi-
cally with well-established, generic CC strategies to equip computational systems with creative responsibilities. Our framework differs from AutoML not only in its stronger focus on generative models, but also in the assumed goals of the generative DL pipeline. More specifically, we identify targets for automation based on the wide and successful application of generative DL in artistic work. In contrast to standard applications, artistic ML engineers and users aim to produce artefacts of high cultural value over perfectly generalised reproductions of the training data.

Our main contribution is to gather, standardise and highlight opportunities to automate generative DL for artistic applications. We identify commonalities of DL pipelines in artistic projects and bring them together in a common framework. This provides a starting point for handing over creative responsibilities in a range of applications, not only artistic. We concentrate our efforts on generative deep learning, rather than generative ML more generally. While we assume the majority of applications to be built on DL approaches, we do not rule out that other generative ML methods might be used within the framework. Our contribution does not consist of a formal solution to a singular automation problem. In contrast, we aim to provide a big picture view of all automation tasks and their associated opportunities and challenges, to be solved in future work.

To leverage insights from CC in the development of our framework, we first clarify the relationship between automating generative DL and endowing artificial systems with creative responsibility. We then outline a standard non-automated pipeline for the development and deployment of generative deep learning models, and show how applications in artistic settings differ from this standard pipeline. Drawing from these two sources, we lay out the automated generative deep learning pipeline, describe several targets for automation therein and suggest ways in which automation could be achieved. We continue with an illustrative example to demonstrate how our framework can give inspiration and guidance in the process of gradually handing over creative responsibility to a generative system. We analyse the relationship between automation and creative autonomy in the context of our framework. We conclude the paper by discussing the limitations of our framework and suggest directions for future work.

**Automated, Artistic Deep Learning as Co-Creation**

We believe that the development of a framework for automated generative DL can benefit from the insights gathered over more than two decades of CC research, because the automation of targets in generative DL can be considered a specific instance of the grand CC goal to give computational systems responsibility over decisions in a creative process.

With each creative responsibility that is handed over to the system, i.e. with each target that is being automated, we increase the computational system’s creative autonomy (Jennings, 2010; McCormack, Gifford, and Hutchings, 2019; Guckelsberger, Salge, and Colton, 2017), i.e. its capacity to operate independently of a human instructor, allowing for it to be ultimately considered a creator in its own right (Colton, 2008b). Crucially though, the users of automated generative DL typically want to retain some control over the automation and its outcome. In developing our framework, we must thus decide which responsibilities should be retained in order to sustain certain modes of interaction between the artistic users and the generative DL system.

To this end, it is useful to frame this interaction in the process of automation as a co-creative act. We adopt Kantosalo et al.’s (2014) working definition of human-computer co-creativity as “collaborative creativity where both the human and the computer take creative responsibility for the generation of a creative artefact”. To qualify as a collaborative activity, both human and system must achieve shared goals (Kantosalo et al., 2014, drawing on Terveen, 1995).

Different automation strategies can enable two coarse forms of interaction. First, the user and system could engage in task-divided co-creativity, in which “co-creative partners take specific roles within the co-creative process, producing new concepts satisfying the requirements of one party” (Kantosalo and Toivonen, 2016). Second, they could engage in alternating co-creativity, where both partners “take turns in creating a new concept satisfying the requirements of both parties” (Kantosalo and Toivonen, 2016).

Alternating co-creativity requires the computational system to not only exhibit creative responsibility for either the generation or evaluation of artefacts, but for both. Crucially, even a non-automated generative DL system can be considered creative in a minimal sense, in that it (despite the name) not only “merely generates” (Ventura, 2016) new samples or artefacts, but also evaluates their proximity to the training set via its loss function. This is accomplished either explicitly, through likelihood estimation, or implicitly, with the help of a critic in an adversarial setting. The system thus produces artefacts that are novel and valuable, realising both requirements of the two-component standard definition of creativity (Runco and Jaeger, 2012). We write “creative in a minimal sense”, because the novelty of artefacts will decline, while their value increases, the better the system approximates the (unknown) distribution from which the training data was drawn.

The definition of the training set and loss function by the user satisfies that both partners interact towards shared goals. Through different ways to automate the ML pipeline, we can free the human partner from certain manual work, while retaining specific creative responsibilities.

We believe that providing the computational system with creative responsibility in the form of automating certain targets does not constrain, but rather expands the shared creative process. The person or producer has, due to their personality and cognitive characteristics, a strong impact on the creative process, product, and the creative environment, i.e. the press (Rhodes, 1961; Jordanous, 2016). However, human creativity is also limited, e.g. due to our bounded rationality (Simon, 1990). A computational system can complement human shortcomings, e.g. via its higher information processing or memory capacity, enabling creativity on larger search spaces (Boden, 2003; Wiggins, 2006).
A Standard Generative Pipeline and Artistic Deviations

We outline the various steps in the process of building and deploying a generative DL model for standard non-automated usage and contrast it with the particular differences that arise when using a model in different artistic contexts. Additionally, we provide a brief overview post-training modifications that aim for active divergence (Berns and Colton, 2020), allowing to manipulate a model into producing artefacts that do not exactly resemble the training data. A more detailed survey of such techniques can be found in Broad et al. (2021). Our goal is to highlight the many choices that have to be taken in the construction of a generative DL pipeline and identify those tasks which pose an opportunity for automation in our framework.

Data Acquisition

The first step towards developing generative models is data acquisition. We distinguish two cases: (i) using pre-existing data sets and (ii) creating new ones. It should be noted, that generative ML is also applied in privacy sensitive areas such as medicine, and in the augmentation of small data sets, as it can produce synthetic data to replace an entire data set or supplement it with additional samples. The augmentation by way of a generative model can be necessary whenever a data set is too small to train another model (e.g. a classifier) with a high number of parameters (i.e. weights and biases in a neural network). However, when the generative model itself requires a large amount of training data, other pre-training data augmentation steps through graphic manipulations can help to do so effectively (Karras et al., 2020).

Using Existing Data Sets In a research setting, it is most common to use standard benchmark data sets or subsets thereof, for training and evaluating generative models. It is generally best practice in machine learning to split the data into training, test and validation subsets. However, generative models are sometimes trained on the entire data set and alternative methods of evaluation are used.

Creating a New Data Set When creating a data set from scratch, the goal is normally to fully represent the subject or category that is being modelled. Therefore, as much data as possible will be collected to maximise variation in the data set and to represent all modes as evenly as possible, i.e. the variety of artefacts that are statistically significantly different from one another. Creating varied, high-quality data sets with the large amounts of data required for training generative models can be very labour intensive and usually the purview of a select few academic and industry laboratories. This is often performed in a distributed fashion, where many workers are involved in collecting, evaluating and labelling data samples.

In contrast to data sets created for industrial and research applications, data sets for artistic purposes are often composed with very different goals. It may not be important to accurately and fully represent a subject matter or domain, as long as the end goal produces interesting results. Data sets are often much smaller, and the considerations for the desired aesthetic characteristics in the end results are much more important in deciding which examples should and which should not be included in the data set. A lot of effort will go into sourcing material and the resulting data sets are much more likely to reflect an artist’s individual style and (visual) language. In some cases, the entire data set will come from an artist’s personal archive (Ridler, 2017).

Training

The objective of training a generative model is to learn a mapping function from an easily controllable and well understood distribution, e.g. a standard Gaussian, to a distribution of much higher complexity and dimensionality, e.g. that of natural colour images. There are a number of different training schemes, which apply to different architectures. They are commonly categorised by their formulation of the training objective. Methods maximise the likelihood of the data either (i) explicitly, such as auto-regressive and flow-based models, (ii) approximately, e.g. variational autoencoders (VAEs), or (iii) implicitly (GANs). When using a method that explicitly models the data, training will be performed until a desired likelihood score is reached. With VAEs, the goal of training is to maximise the log-likelihood of the data set. In the adversarial setup, the decision when to stop training is less clear. Training is often run for a pre-specified period and the results are evaluated qualitatively. A fully trained model ideally represents the entire training data distribution, and can be sampled randomly to produce good results. Another desirable quality is that interpolation between two input vectors is matched in the outputs.

Generalisation is a goal of almost all ML systems and applications. A model should be able to generalise to unseen data, while not underfitting or overfitting the training data. In an artistic setting, however, this is often less important, and if it produces interesting results, artists may often embrace the aesthetic qualities of an underfit (Shane, 2018) or overfit model (Broad and Grierson, 2017).

Evaluation

The general performance of a model is measured in terms of the distance of the learned distribution to the target distribution. A model further ideally covers all modes in the input data set. For generative methods that explicitly model a probability distribution over the data, the (log) likelihood can be measured and evaluated directly. Implicit methods, such as GANs, have to be assessed with other metrics such as the Inception Score (Salimans et al., 2016) and the Fréchet Inception Distance (FID) (Heusel et al., 2017). As these metrics are only a simplified standard for evaluation and have some shortcomings, additional qualitative checks might be needed to ensure fidelity of the output.

While in some artistic settings good quantitative performance might matter, it can be ignored entirely in others, and a qualitative assessment of the output is usually much more important. Quality, diversity and accuracy may not be the only considerations (and may even be actively avoided), whereas novelty, interesting mis-representations of the data and other aesthetic qualities may be desired. Due to the variety of qualities that an artist might look for in a model’s
output, there is no unique or widely used standard metric for evaluation. This is rooted in the highly individualistic nature of artistic work and linked to the additional strategies for iterative improvements and curation of the output which we discuss in the following subsections.

Iterative Improvements of Outputs
Here we look at the diverging strategies for the gradual improvement of a system’s output in a research and development versus an artistic setting.

Iterating on the Model In the research and development of generative models, the data set often remains fixed, while various aspects of the network architecture and training regime will be altered. For instance, various optimisation hyper-parameters will be evaluated, such as: learning rate, momentum or batch size; or network configurations: number of layers, type of activation functions, etc. Different training regimes may also be experimented with, such as: optimisation algorithms, loss functions, and methods for regularisation and sampling.

Iterating on the Data Set In artistic contexts, it is much more common to iterate on the data set and keep other parameters fixed, before possibly making iterative improvements to the network and model parameters. Data that appears to be producing unwanted results, or skewing the model in certain directions may be removed. Revisiting the composition of samples (such as cropping), and the removal and addition of samples in order to refine the data set may be undertaken (Schultz, 2020).

Deployment
Generative models are used differently in standard and artistic settings in accordance with their respective goal. We here differentiate between standard sampling and output curation.

Standard Sampling Generative models are trained with the goal that they can be sampled randomly and every generated output will be of value and high typicality (Ritchie, 2007). Therefore, in most standard applications models are simply sampled randomly with no additional filtering taking place. When filtering is performed, it is often done with the goal of quality evaluation, such as using the discriminator for evaluation quality (Azadi et al., 2019), or using the Contrastive Language-Image Pretraining (CLIP) model (Radford et al., 2021), as was the case in evaluating and ranking the generated outputs of the discrete VAE model in the DALL-E image generation project (Ramesh et al., 2021).

Output Curation Rather than sampling randomly from a model, artists will often spend a lot of time curating a model’s output. The goal of building a model in an artistic setting is not necessarily to generate only samples of high value, but to produce some interesting or novel results, which can then be hand-selected. This can be through filtering samples or searching and exploring the latent space. In some cases, such as combining language-image models with latent space search for text-to-image generation, e.g. Murdoch (2021), much effort goes into prompt engineering to find a specific latent vector that produces interesting results.

Post-training Modifications
Having looked previously at the curation of a model’s output in an artistic setting, i.e. the act of identifying the few artefacts of interest in a large set of output samples, we now turn to active divergence techniques (Berns and Colton, 2020) which aim at consistently producing results that diverge from the training data. These strategies, specifically developed in creative contexts for the purpose of art production, include hacks, tricks and modifications to the model parameters, as well as the daisy-chaining of several models.

One approach is to find a set of parameters where the generated artefacts blend characteristics of multiple data sets. For this, a pre-trained model can be fine-tuned on a second data set, different from the original data. As soon as the results present an optimal blend between the two data domains, the fine-tuning can be stopped. This mixture of data sets can also be achieved by blending the weights of two models. Either interpolating on the weight parameters of the two models, or swapping layers between models, so that the new model contains higher level characteristics of one model, and lower level characteristics of another. Another method consists in chaining multiple models together. This allows artists to explore and combine characteristics of different data sets. Unconditional generative models will often be chained together with domain-translation models, e.g. CycleGAN (Zhu et al., 2017) for sketch-to-image translation, or style transfer algorithms (Gatys, Ecker, and Bethge, 2016). The aim of such pipelines is to produce artefacts that reflect the complex combination of characteristics from many data sets.

Another approach is to modify the model in order to have artefacts completely diverge from any training data. An existing pre-trained model can be fine-tuned using a loss function that maximises the likelihood over the training data (Broad, Leymarie, and Grierson, 2020). Other techniques intelligently combine learned features across various models (Guzdial and Riedl, 2018), or rewrite the weights of the model (Bau et al., 2020), re-configuring them to represent novel data categories or semantic relationships. In contrast, network bending does not require any changes to the weights of the model (Broad, Leymarie, and Grierson, 2021). An analysis of the model is performed to determine which features are responsible for generating different semantic properties in the generated output. Deterministically controlled filters are then inserted as new layers into a model and applied to the activation maps of features.

An Automation Framework
We build our framework drawing on the standard generative DL pipeline and its artistic deviations, as previously described. We first discuss automation as a search problem and the various techniques it can be approached. We then go on to list the targets for automation in a generative deep learning pipeline for artistic purposes. Some decisions have implications for other targets further down the line, e.g. the number and type of hyper-parameters depend in part on the kind of network architecture and optimisation...
algorithm. While the process is presented as a sequence of consecutive steps from input to output, it should be understood that all steps are optional and flexibility is required. Improving a system’s output works best as an iterative loop in which we might go back and adjust or intervene at any given prior step.

We define the terminology of our framework as follows. With automation, we refer to the act of addressing with computational means those decisions in a generative deep learning pipeline that normally would be taken by a person. A target is defined as one such decision which provides an opportunity for automated instead of manual tuning.

**Automation as a Search Problem**

A generative pipeline is automated by assigning responsibilities over individual targets to either the user or the system. While those retained by a person will have to be tuned manually, all other targets require the system to determine a configuration independently. This problem is analogous to the search problem over hyper-parameters in AutoML. The possible values of each automated target effectively construct a search space over possible system configurations. The number of total permutations, and the resulting search space, can grow rapidly with every independent target added.

Limiting continuous parameter values to a reduced range or a set of discrete values, as per grid search for machine learning hyper-parameters, can help make the problem more feasible. The formulation as a search problem is the standard way to tackle automation in AutoML. However, extensive search over meta-parameters can be computationally expensive, time-consuming, cause high energy consumption and consequently have a considerable environmental impact.

The extensive work on search problems provides numerous approaches to constrain this search. Strategies range from complete, to informed, to random methods. While exhaustive search can yield an optimal solution, it can be impractical and often infeasible for large search spaces. Random sampling, on the other extreme, can be a surprisingly effective strategy at low cost and with potentially surprising results. While Jennings (2010) requires a system to meet the non-randomness criterion in order to be considered creatively autonomous, this definition does not rule out all uses of randomness and allows for testing random perturbations to a system’s standards. AI-based search methods can benefit from meaningful heuristics and leverage both exploration and exploitation (e.g. evolutionary search). Gradient-based methods have seen a lot of progress in recent years. Other approaches include rule-based selection and expert systems, with drawbacks including that they require manual construction and expert knowledge.

Finally, machine learning itself can be used to choose values through a pre-trained model. Indeed, practitioners in generative deep learning tend to go directly to automation via deep learning. In particular, recent advances in contrastive language-image pre-training (Radford et al., 2021) allow for computing similarities between text and images. Such a model could take over the responsibility of assessing whether an image looks like a text description, or vice versa, at any point in the pipeline where a human artist would do the same task. All of the above approaches can be applied in an iterative fashion over subsets of the search space, gradually limiting the range of possible values.

**Automation vs. Autonomy**

While we have primarily focused on increasing a system’s creative autonomy through automation, our framework does not grant a system as much autonomy as to enable it to act entirely independently in response to its own motivations (cf. Guckelsberger, Salge, and Colton, 2017). A system within our framework would remain inactive until engaged with. Such engagement can range from a stimulus through available sensors, e.g. cameras, microphones or heat sensors, to a text or image prompt or an entire inspiring set (Ritchie, 2007), to more precise and detailed instructions. In any case, this choice of input channel and sensibility has to be taken by a human and is not a target in our framework.
We further assume the choice of generated media (image, audio, text, video, etc.) to be made by a person prior to building a system. Naturally, it is not difficult to imagine a setup in which this choice, too, becomes part of the pipeline. Going one step further in autonomous automation, our framework and its targets make it possible to devise a generative system which produces automated generative pipelines. In fact, it might be possible for a generative system to generate itself, much like a general-purpose compiler that compiles its own source code. This self-referential generation has similarly been proposed in work on automated process invention (Charnley, Colton, and Llano, 2014).

**Targets for Automation**

Below we define and discuss the many tasks and decisions that are part of a generative DL pipeline in an artistic setting and which can be automated within our framework. Whenever applicable, we explain how a target relates to concepts of AutoML and CC.

The following subsections identify individual targets for automation. The complete process is illustrated as a sequence of steps in figure 1. As per this diagram, we organise the steps into three stages: (i) a *preparation* stage to gather relevant materials (ii) a *configuration* stage, where the models, training regimes and parameters are tuned to produce valuable output, and (iii) a *presentation* stage where the user deploys a final model and curates the output. The first target (selecting a pre-trained model) is optional and can be skipped in order to start from scratch instead. In this case, we begin with data preparation and curation.

**Pre-trained model (optional)** It might not be necessary to train a network from scratch if an appropriate pre-trained model is available, especially when a quick system setup is desired. A list of pre-trained models, tagged with keywords associated to their generative domain, could provide a knowledge base for a system to select, download and deploy a model. This can either be directly put to use, in which case the system could immediately skip to evaluating the model, or it can be fine-tuned on a smaller set of data. Such additional fine-tuning could be dependent on the outcome of the pre-trained model’s evaluation. Only if the pre-trained model’s output is not satisfactory would it have to be further optimised or de-optimised. Working with a pre-trained model has implications for the subsequent choices of the network architecture, training scheme and loss function.

**Data preparation and curation** This preparation step includes the acquisition, cleaning, augmentation and transformation of data samples, akin to data preparation in AutoML. Starting with the data collection task, we consider different data sources from which a system could select. Drawing on existing data sets, such as an artist’s private data collection, can introduce important desirable biases and ensure high quality output. In contrast, scraping samples from the internet could contribute to the generation of surprising results. Additional pre-trained generative models can provide a source for synthesised data in large quantities.

An important addition to the pre-processing is data curation, in contrast to simple cleaning. Rather than filtering out noisy samples, for artistic purposes it can be desirable to add ‘noise’. To this end, it is not uncommon in an artistic context to mix multiple data sets. In this additional step, the system further refines the data set, similar to an artist adding or removing individual samples, which can influence the quality of the system’s final output. This is an opportunity for iterative improvements and for *alternating co-creativity* (Kantosalo and Toivonen, 2016), given that the system both generates and evaluates. Automation in the cleaning and curation tasks can be achieved, e.g. in the image domain, by employing other computer vision or contrastive language-image models.

**Network architecture and training scheme** This target for automation defines the choice of possible architectures (e.g. GAN, VAE, Transformer), which could include non-neural methods. Neural architecture search (NAS) in AutoML is concerned with finding optimal combinations of basic building blocks of artificial neural networks in terms of performance on a classification or regression task, an immensely difficult optimisation problem. We recommend in our framework to instead select from tried-and-tested architectures, only altering parts of the architecture with a direct influence on the output, e.g. the number of upsampling convolutions which determine the final output image size.

The training scheme is largely influenced by the choice of architecture. In the case of GANs, the training scheme includes the choice of whether to train the discriminator and generator networks in parallel or consecutively, and how many individual optimisation steps to perform for either.

**Loss function** The formulation of the basic loss term is highly dependent on a model’s training scheme and constitutes the minimum requirement for successful training. However, additional loss terms can change or supplement the basic term for further refinement of the training objective. As a central part in guiding the model parameter optimisation process, any modification to the loss terms will strongly impact the modelled distribution and consequently the system’s output. In other contexts, methods have been proposed for the automatic invention of objective functions (Colton, 2008a). These could provide a starting point for adapting the approach to the constraints of loss functions in generative DL.

**Optimisation algorithm** The selected algorithm will be responsible for adjusting a model’s parameters through error correction informed by the gradient of the loss function. This choice can potentially have an influence on the system’s output, as it is responsible for finding one of the potentially many local minima in the loss landscape. As it determines whether convergence can be reached at all, this decision can ultimately make or break the success of the training process. It can further largely influence convergence speed and be critical in time-sensitive setups. The choice of optimisation algorithms might be limited by the previous selection of network architecture and corresponding training scheme.

**Hyper-parameter tuning** Optimisation of batch size, learning rate, momentum, etc. can be achieved via AutoML methods, and there is much active research in this area.
Model selection and evaluation  From all the possible models, the best one has to be selected in accordance with given criteria relevant to the task at hand. As the training process is essentially a succession of gradual changes of model parameters over time, this task is equivalent to identifying the right moment to stop training. Additionally, and in order not to lose previous training states, model checkpoints can be saved along the way as training progresses and whenever model evaluation satisfies given criteria. After training is finished, the best model has to be selected from all candidate checkpoints. In standard ML projects, this would normally be done with respect to the primary concern of predictive accuracy. But in generative projects, other considerations may include how surprising the outputs are, synthesis speed (for tool or real-time uses) and coherence of the results. Such criteria could be employed in a weighted sum of metrics, where the system can give more or less emphasis to individual terms. This would allow the combination of standard metrics like FID in the image domain for general output fidelity with a measure for sample similarity compared to a reference sample(s), inspiring set or text prompt via a contrastive language-image model.

Output curation  Having obtained a successfully trained model, we want a system to reliably produce high-quality output. While efforts in previous steps were aimed at refining the model which is at the core of the generative process, this final automation target aims to raise the system’s overall output quality. Two approaches come to mind: filtering and search. In the former, a system could select those samples from a large batch of model outputs that rank highest against a given metric. In the latter, the system could search for vectors directly in a model’s latent space via one of the various methods we have outlined in the section above on approaches to search problems. The evaluation measure, as before, could be the similarity of samples compared to a set of reference samples, an inspiring set or a text prompt via a contrastive language-image model.

An Illustrative Example

In early 2021, a generative deep learning Colab notebook (Bisong, 2019) called the Big Sleep was shared online (Murdock, 2021). It allows for text-to-image generation (Agnese et al., 2020), effectively visualising a user-generated text prompt, often with innovative content and design choices, as per the example in figure 2. This is an instance of an artistic deviation from the standard pipeline, where CLIP (Radford et al., 2021) is used to evaluate a generated image w.r.t. a given text prompt, driving a gradient-based search for latent vector inputs to a generative model called BigGAN (Brock, Donahue, and Simonyan, 2019). We use this setup as an example to identify the following places where automation could be introduced, based on our framework. We highlight concrete techniques and references for automation from the literature.

- In terms of pre-trained model selection, numerous people have substituted BigGAN with other GAN generators. This creative responsibility could be automated, with the system choosing from a database of GANs and installing new ones into the notebook.

- In terms of data preparation and curation, users often choose imaginative text prompts, as the notebook often produces high quality, surprising results for these. This could be substituted, for example, with automated fictional ideation techniques (Llano et al., 2016).

- Murdock (2021), the notebook programmer, innovated in loss function definition, employing patches from generated images rather than the entire image to evaluate its fit to the prompt. Various image manipulation routines could be automatically tested within loss function calculations from a library, with the system automatically altering the notebook at code level.

- As described in Colton et al. (2021), in some circumstances where multiple images are being generated simultaneously, increasing the learning rate can help searches fail quickly. Such hyper-parameter tuning could be automated using standard AutoML techniques, guided by requirements on acceptable search successes and output image quality.

- In terms of model selection and deployment, we can imagine models being used as creative web-services (Veale, 2013), with higher-level CC systems accessing text-to-image generators in a variety of projects.

- There has been an explosion of human usage of notebooks like the Big Sleep, with attendant output curation via cherry picking results for posting on social networks and in blogs. This would be an ideal target for automation with systems using CLIP and other techniques to evaluate images, also possibly inventing new aesthetic measures (Colton, 2008a).

Discussion

We have presented a framework for the specific purpose of automating manual tasks in a generative DL pipeline for artistic projects. We adopt the core concepts of AutoML and adjust them in two ways. First, we focus on generative DL which differs in the type of learning task, in that it is concerned with modelling the distribution of a training set, rather than classification or regression. And second, we address the artistic usage of generative DL, where more emphasis is given to the qualities of the generated output over the qualities of the model. The specialisation of our framework inversely limits its generalisability in the same ways. On the one hand, there might be artefact-driven applications of generative DL within or outside CC that we have not con-
sidered. On the other hand, our framework is not generally applicable to generative approaches in DL due to its special emphasis on artistic uses. Its focus on generative DL further limits its validity for other generative modelling methods.

**Automation and Creative Autonomy**

We have previously analysed the close relationship between the automation of generative DL systems and the central CC goal to increase a system’s creative autonomy (Jennings, 2010; McCormack, Gifford, and Hutchings, 2019; Guckelsberger, Salge, and Colton, 2017) by granting it more creative responsibilities (Colton, 2008b). Here, we complement this *a priori* analysis based on knowledge of our concrete automation pipeline. The aim is to understand to which extent our proposed pipeline already enables facets of creative autonomy, and how CC insights on creative autonomy could be used to advance it in future work.

Automation is necessary for creative autonomy, but the opposite does not hold: while a fully automated generative DL system might still exactly follow user-prescribed goals, an autonomously creative system has the “freedom to pursue a course independent of its programmer’s or operator’s intentions” (Jennings, 2010). This firstly requires the system to autonomously evaluate its creations, which is satisfied by any system that can be considered creative (Ventura, 2016). In addition, an autonomously creative system must be capable of autonomous change, i.e. initiating and guiding changes to its standards without being explicitly directed when and how to do so” (Jennings, 2010). To prevent trivial implementations of these capabilities, Jennings requires them to not exclusively rely on random decisions.

To assess how much our pipeline realises creative autonomy, we can draw on various CC approaches to enhancing autonomy in computational systems. For instance, Colton (2009) proposes “repeatedly asking ourselves: what am I using the software for now? Once we identify why we are using the software, we can […] write code that allows the software to use itself for the same purpose. If we can repeatedly ask, answer and code for these questions, the software will eventually […] create autonomously for a purpose, with no human involvement”. Our framework provides various candidate targets to perform such a gradual elevation of a generative DL system.

For the evaluation of a concrete system built under our framework, we consider the FACE model (Colton, Charnley, and Pease, 2011; Pease and Colton, 2011) an adequate evaluation tool. In this evaluation model, systems are described in terms of the creative acts they perform. Such an analysis allows for the identification of newly added creative responsibilities through automation.

Linkola et al. (2017) follow a more constrained approach and, as part of a larger agenda to realise meta-creativity in CC, propose that creative autonomy requires artefact-, goal- and potentially generator-awareness, realised through operators of (self-) reflection and (self-) control which closely match Jennings’ (2010) requirements for evaluation and change. Whether a system built within our framework satisfies these definitions depends on the extent to which it is granted responsibilities in the form of automating decision-making for targets identified in the framework. We demonstrate this based on extensions to a non-automated generative DL system. Such a system can be considered to have some generator-awareness due to the role of its loss function (self-reflection), and its adjustments of own parameters through error correction methods like back-propagation (weak self-control). A system’s control over changes to its generator can be increased from weak to strong within our framework, through the automated manipulation of network architecture or selection of a pre-trained model. Further putting a system in charge of its loss function within our framework (strong control) affords it goal-awareness and consideration as autonomously creative, if it is capable of modifying the loss function in response to its evaluation of generated output.

Crucially, more radical forms of creative autonomy do not eliminate co-creation, i.e. cut ties with the system user entirely, but facilitate different forms of interaction. To really become independent of its designer, a system must not be isolated but interact with critics and creators that shape its evaluation and changes (Jennings, 2010). A fully creatively autonomous system might refuse the will of its interaction partner (Jennings, 2010; Guckelsberger, Salge, and Colton, 2017), but we believe that this holds a promise for innovative artistic collaborations between people and computational systems, connecting artistic practices in generative DL with the philosophy and goals of CC.

**Future Work**

In this formulation of our framework we have only briefly mentioned automation of creative responsibilities via the usage of ML models. The possibility of training or deploying multiple models in the same system enables the addition of organisational structures to our framework, in which we think of individual models as agents in a multi-agent system.

To use our framework in co-creative applications, augmenting a system with the ability to communicate its adjustments and intentions would be especially beneficial. Moreover, to address our framework’s limitations, further work is needed to consider applications which use generative DL but are not artistically focused. This could potentially inform a more general automated ML framework, which would further benefit from more formal definitions.

We plan further study of the ways in which deep learning researchers, practitioners and artists work with generative systems, in particular where they have, and could, add levels of automation, via analyses such as the illustrative example above. Some of the techniques that artists apply, such as data set curation and iteration, as well as the selection of generated outputs, are promising avenues for automation and require further investigation. We further plan to put our framework to use in applied projects. Through this, we aim to provide demonstrative examples of how some of the challenges in automation can be tackled and to show the surprising results that automation can afford. For the evaluation of such demonstrative examples we plan to draw from the FACE descriptive model of creative acts (Colton, Charnley, and Pease, 2011; Pease and Colton, 2011).
Acknowledgements

We thank our reviewers for their helpful comments. Sebastian Berns and Terence Broad are funded by the EPSRC Centre for Doctoral Training in Intelligent Games & Games Intelligence (IGGI) [EP/L015846/1, EP/S022325/1]. Christian Guckelsberger is supported by the Academy of Finland Flagship programme “Finnish Center for Artificial Intelligence” (FCAI).

References


Brock, A.; Donahue, J.; and Simonyan, K. 2019. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In Proc. ICLR.


Colton, S. 2008a. Automatic Invention of Fitness Functions with Application to Scene Generation. In Workshops on Applications of Evolutionary Computation.


Guzdial, M., and Riedl, M. O. 2018. Combinets: Creativity via Recombination of Neural Networks. Proc. ICCC.


Entrepreneurship: A New Frontier in a Computational Science of Creativity

Ashok Goel, Mukundan Kuthalam, Sung Jae Hong, Keith McGregor
Design Intelligence Laboratory, School of Interactive Computing,
Georgia Institute of Technology, Atlanta, USA

Abstract
We present entrepreneurship as a new frontier for developing a computational science of creativity. Entrepreneurship often requires a large upfront investment, but customer feedback typically is delayed, and while the costs of failure can be considerable, the gains from success too can be significant. Given its importance, how can we help novice entrepreneurs learn about entrepreneurship? We analyze the role human coaches play in mentoring novice entrepreneurs by asking critical questions to help generate business models. We propose that virtual coaches may augment the learning of novice entrepreneurs. We describe a preliminary experiment in designing a virtual coach named Errol for learning about entrepreneurship. When a startup team creates an initial business model, Errol uses semantic and lexical analyses to ask questions about the model, leading to model revision. Our experiment indicates that creativity may emerge out of the interactions between the virtual coach and a startup team, and that this human-computer co-creativity may accelerate the process by which a novice entrepreneur can learn to create intermediate-level business models.

Keywords: Entrepreneurship, Human-Computer Co-Creativity, Question Asking, Virtual Coaches.

Introduction
The field of computational creativity seeks to develop a computational science of creativity, including computational theories of creativity, techniques for realizing artificial creativity, and tools for supporting human creativity. It addresses creativity in a variety of domains such as the arts, design, literature, science, etc. Our first goal in this position paper is to introduce entrepreneurship as a new frontier for research on computational creativity. Entrepreneurship often requires a large upfront investment, the evaluation through customer feedback typically is delayed, the chances for success are low, and the costs of failure can be high. Yet, when an enterprise succeeds, it can make a difference in the world and the gains can be significant.

Given the importance of entrepreneurship to the economy, the workforce, and the society, many academic institutions around the world are expanding educational programs in creating startups. Thus, our second goal in this article is to present a high-level information-processing account of teaching and learning about entrepreneurship in which experienced serial entrepreneurs act as coaches to novice entrepreneurs. For example, the US NSF’s Innovation Corps (I-Corps) (Huang-Saad, et al. 2017) runs educational programs for scientists and engineers to learn how to instigate such startups. In these programs, human coaches teach fundamentals of entrepreneurship such as customer discovery and business model generation. Much of the teaching uses the Socratic method in which the coaches ask critical questions to encourage reflection by the novice entrepreneurs. Based in part on the questions asked by coaches, novice entrepreneurs learn to build better business models and often make a pivot in their value propositions and customer segments. The creativity of the business models emerges out of these interactions between the coaches and the novice entrepreneurs.

Third, we posit that virtual coaches may amplify the reach of human coaches and augment the learning of novice entrepreneurs. Serial entrepreneurs who are willing to act as coaches to novice entrepreneurs are not easy to find. Further, human coaches often have biases. Given that the questions human coaches ask in early phases of business model generation tend to have well-established patterns, a virtual coach may ask similar questions based on commonly accepted norms. This may help reduce bias while accelerating the process of learning to generate intermediate-level business models. Virtual coaches have the added benefits that they can be used by anyone, anytime, anyplace. Thus, they can support learning how to create a startup on demand and at scale, and thereby help foster a culture of entrepreneurship.

Finally, we briefly describe a preliminary experiment in developing a “proof-of-concept” virtual coach named Errol for teaching entrepreneurship. When a startup team creates an initial business model, Errol uses semantic and lexical analyses of the entries in the model to ask questions. This leads to reflection by the startup team, and results in iterative revision and refinement of the model. By attempting to categorize
and correct the errors that novices typically make, Errol seeks to accelerate the process by which a novice startup team can start creating intermediate-level business models. Errol is an experiment in realizing human-computer co-creativity in entrepreneurship with the creativity of the business models emerging out of its interactions with novice entrepreneurs.

**The Domain of Entrepreneurship**

There is broad consensus in research on computational creativity on several aspects of creativity. For example, creativity requires both a producer and a receiver; creative products are novel, useful and non-obvious; and creativity lies on a spectrum (e.g., Besold, Schorlemmer, and Smaill 2015; Boden 1990; Sternberg 1999; Veale and Cardoso 2020). However, research on computational creativity continues to explore new domains of creativity that are often characterized by new dimensions of the creativity spectrum. For example, in early research on AI in design, Goel and Chandrasekaran (1991) viewed design as a case-based process. They described a spectrum of creativity that starts at one end with the solution to a new design problem being identical to that for a previous problem; moves to the solution to the new design problem differing from that of a previous problem only in the values of parameters of a component; then to the solution to the new design problem differing from that of a previous problem through the substitution of one component by another; next to the solution to the new problem differing from that of a previous problem through the addition or deletion of a component, and so on. In contrast, more recently Fitzgerald, Goel, and Thomaz (2017) have viewed robot creativity as a process of embodied analogical reasoning. They describe a different spectrum of creativity that starts on one end with the new task being identical to a task familiar to the robot; moves to the new task differing from the familiar task in the parameters of an object; then to the new task differing from the familiar task in the parameters of the object itself; next to the new task differing from the familiar task in the relationships among the objects, and so on.

Characterization of creativity in literature and the arts typically requires consideration of additional dimensions (Candy and Edmonds 2002; Cohen 1995; Cope 2005; Pearce and Wiggins 2014; Perez y Perez and Sharples 2001; Veale 2012). Such additional dimensions may pertain to the problem definition because the problem may be ill-defined, and/or the evaluation criteria for the criteria for evaluation may be unknown in advance or difficult to operationalize. The point here again is that each new domain of creativity appears to introduce new dimensions of the spectrum of creativity.

We posit entrepreneurship as a new frontier in creativity, one that offers new challenges both for building a computational theory of creativity and for developing interactive tools for supporting human creativity. As with creativity in general, creativity in entrepreneurship requires both a producer and a receiver, and the products of creativity are at least initially novel, useful, and non-obvious to the intended receiver. Further as with creativity in literature and the arts, problems in entrepreneurship are ill-defined and the evaluation criteria are unknown in advance or difficult to operationalize.

In addition, creativity in entrepreneurship is defined by a large upfront financial investment and much delayed evaluation through customer feedback. Further, unlike many other creative domains, the chances of success in entrepreneurship typically are low and the economic costs of failure can be high. However, when an enterprise succeeds, it can make a real difference to the lives of the receivers and result in the significant financial gains to the producers as well as economic benefits to the community of producers and receivers as a whole. These additional dimensions make entrepreneurship a unique and challenging domain because the margin for errors is very small - much smaller than in many other domains of creativity.

**Learning about Entrepreneurship**

Entrepreneurship is critical not only to employment, but also to the economy and the society of a country (Carland & Carland 2004). Thus, many institutions around the world are expanding and strengthening their educational programs in entrepreneurship (Kuratko 2004) such as the CREATE-X program at Georgia Tech (Forest et al. 2021). We will use the I-Corps program for academic scientists and engineers as a case study to develop a high-level information-processing account of the teaching and the learning in entrepreneurship education.

The I-Corps program focuses on startup creation by bringing the rigor of engineering science to entrepreneurship. Thus, the novice entrepreneurs are encouraged to think of entrepreneurship in terms of the scientific process: hypothesis generation, experiment design, data collection, hypothesis revision and refinement, and so on. A startup’s business model articulates and elaborates on its business hypotheses. In the I-Corps program, serial entrepreneurs act as human coaches to novice entrepreneurs, teaching fundamentals of entrepreneurship such as customer discovery and business model generation. Much of the teaching uses the Socratic method (Paul & Elder 2007) in which the coaches rarely give direct advice; instead, they ask critical questions to encourage reflection by the novice entrepreneurs. Based in part on the questions asked by coaches, novice entrepreneurs learn to build better business models and often revise their hypotheses by making a pivot in their value propositions and customer segments. The creativity of the business models
emerges out of the interactions between the coaches and the startup team.

**Business Model Canvas**

The I-Corps program uses Osterwalder & Pigneur's (2010) Business Model Canvas (BMC) for articulating and elaborating a startup's business hypotheses. A BMC is a modeling template that simplifies the description of a business into three major categories and nine sections within them: Desirability (Customer Segments, Value Propositions, Customer Relationships, Channels), Feasibility (Key Activities, Key Resources, Key Partners), and Viability (Cost Structure, Revenue Streams). Figure 1 provides a snapshot of an evolving business model.

The BMC affords would-be entrepreneurs cognitive support for articulating, elaborating, sharing, and critiquing their hypotheses about the value their potential business adds to some market, the targeted customer segments, and several other components necessary to spawn their startups. Yet developing a good business model using a BMC also requires getting thorough and highly specialized feedback on what is good and what can be improved. Thus, I-Corps coaches provide iterative feedback on a startup’s BMC, with the simultaneous intent of enlightening the novice entrepreneur as well as enabling them to provide the right products and services to the right markets, thereby optimizing their potential for success.

**Human Coaching**

We have extensively consulted with some of the human coaches in the I-Corps program; indeed, the last author on this paper (KM) is a lead instructor in the program. We have also observed human coaching in the I-Corps program in practice; the first author (AG) has twice taken the program to learn about spawning startups. In addition, we have analyzed coaches’ interactions with novice entrepreneurs in the teaching of the I-Corps Puerto Rico cohort in 2018. Our analysis indicates that the coaches simultaneously are trying to understand the developing business model on a given BMC as a whole and determining what the various segments on the BMC are lacking in content, structure, and relationships. We found that coaches then formulated questions designed to help the novice entrepreneurs become aware of the relationships described on a BMC.
into classes of errors to help the novice entrepreneurs become aware of the errors and encourage them to reflect on and seek to address the errors.

While the I-Corps program may be considered a success as evidenced by its rapid expansion over the last decade (Huang-Saad et al. 2017), education in entrepreneurship in general faces several obstacles. The first problem is scaling up the teaching and learning in the I-Corps program to other formal and informal programs in entrepreneurship education. However, serial entrepreneurs who can act as human coaches and provide feedback on business models of novice entrepreneurs are not easy to find. Thus, scaling the teaching and learning in I-Corps to other entrepreneurship programs and startup incubators is difficult. Additionally, even within an educational program, on-demand access to human coaches is limited because of constraints on their available time. Thus, from the perspective of novice entrepreneurs, they do not have adequate access to human coaches who would give them critical feedback on their business models, affording faster improvements to their business modeling practices.

Further, the teaching skills required to provide feedback that strictly adheres to the Socratic method of critical questioning without offering consultative advice is not very common among human coaches. Finally, human coaches have not only many of the typical cognitive biases, but also opinions based on their personal experiences with entrepreneurship. These opinions can make their feedback on a business model proposed by a startup team subjective, biased and skewed.

Virtual Coaches

We posit that virtual coaches may amplify the reach of human coaches and augment the learning of the novice entrepreneurs. The early stages of developing a business model follow a repetitive pattern of common errors by most novice entrepreneurs. Thus, human coaches give the same types of feedback to startup after startup, cohort after cohort. Given that the feedback in the early stages tends to have well-established patterns based on commonly accepted norms, a virtual coach may ask similar questions. This may help reduce bias in the feedback while accelerating the process of learning to generate intermediate-level business models. Once a novice startup team has progressed to an intermediate level of expertise, human coaches may take over from the virtual coach and help the team in reaching advanced levels of expertise. Virtual coaches have the added benefits that they can be used by anyone, anytime, anyplace. Thus, they can support

Figure 2: The user interface to Errol. Since Errol currently addresses only value propositions and customer segments, other fields in the BMC have been left blank. The pencil icons allow users to edit entries while the x’s are for deletions. The colors, like [red], indicate an underlying relationship between customer segments and value propositions. In this example, all items tagged [red] are related to each other.
learning about entrepreneurship on demand and at scale.

It is important to note that the virtual coach need not, indeed should not, precisely mimic a human coach; instead, the goal is to simulate the experience of a human coach reviewing a business model represented on a BMC and giving directed feedback by asking pointed questions, thereby providing guidance that novice learners of entrepreneurship may find useful in improving their business model. Instead of mimicking a (possibly biased) human coach, a virtual coach should try to address typical novice errors in business model generation.

More broadly, a virtual coach should act like a “Sounding Board” (Schank & Cleary 1995) to the startup team. The key to addressing learning in an open-ended problem such as design or entrepreneurship is to ask the right questions. The startup team likely will know more about the given problem than its coach. Thus, the startup team should maintain control and the virtual coach should only supply the team with questions that it believes are appropriate to pursue. The virtual coach might not know much about the problem at hand; all it will know are what types of questions are useful to ask and how to present the questions in a sensible order. A passive learner may choose to follow the path suggested by the virtual coach, but a more active learner may make a pivot based in part on the questioning, revise the business model, and thereby change the course of questioning. The creativity in the final business model will emerge from the interactions between the virtual coach and the startup team.

**Errol: Teaching by Asking Questions**

Errol is a preliminary, “proof-of-concept” virtual coach that captures a few of the essential characteristics of an idealized startup virtual coach. Errol focuses on giving feedback by asking questions a business model captured on a BMC, simulating the experience of receiving expert Socratic guidance to improve a business model. Errol takes as input the business model expressed on the BMC and generates questions as output that either ask the learner to clarify the contents of the business model or seek to address the errors common to novice entrepreneurs. By adhering strictly to providing feedback only in the Socratic method, Errol invites the startup team to reflect on its own reasoning and to deliberate on its hypotheses and the assumptions underlying them.

While all nine elements of BMC are relevant to the model of the emerging business, the current version of Errol focuses on the Desirability aspect of the proposed startup, and expressly upon the two most important sections and the relationships between them: Value Propositions and Customer Segments. These two sections are the primary drivers that determine the

---

**Figure 3:** A graphical representation of Errol’s processing. Errol extracts from the user’s business model value propositions, customer segments, and relationships between them, and provides them to both lexical and semantic processing. The semantic processing focuses on the nouns in each of entries, while the lexical processing goes straight to rules.
Questions Generated by Customer Segment Triggers

<table>
<thead>
<tr>
<th>Customer Segments</th>
<th>Generated Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Health Educators, Psychologists, Professional Counselors, Nurses, Pharmacist, Health Administrators</td>
<td>Overall, could you try to be more specific about who you are trying to target with &quot;Health Educators&quot;?</td>
</tr>
<tr>
<td>Health Educators, Psychologists, Professional Counselors, Nurses, Pharmacist, Health Administrators</td>
<td>Can you expand on &quot;Health Educators&quot;?</td>
</tr>
</tbody>
</table>

Questions Generated by Value Proposition Triggers

<table>
<thead>
<tr>
<th>Value Proposition</th>
<th>Generated Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provide assistance with reminders of deadlines, tracking, and monitoring requirements</td>
<td>Would you mind expanding on some of these phrases in &quot;Provide assistance with reminders of deadlines, tracking, and monitoring requirements&quot;?</td>
</tr>
</tbody>
</table>

Remaining Generated Questions

- We know that health is related to being. I would suggest incorporating this idea into your BMC.
- We know that counselors are capable of giving advice. I would suggest incorporating this idea into your BMC.
- We know that nurses are capable of care for patients. I would suggest incorporating this idea into your BMC.
- We know that nurses work at a hospital. Is there anyone else related to this location who would also be interested in your business?
- We know that nurses are at location hospital. I would suggest incorporating this idea into your BMC.
- We know that nurses is a care giver. I would suggest incorporating this idea into your BMC.
- We know that pharmacist is a person. I would suggest incorporating this idea into your BMC.
- We know that health is related to well. I would suggest incorporating this idea into your BMC.
- Is every customer relevant to your business? Have you segmented your customers too much?

Table 1: Questions generated for BMC 709.

Remainder of a business model. Customer Segments (CS) declare which customers will be served by the business while Value Propositions (VP) address what value the business will add to satisfy the needs of those Customer Segments. Figure 2 illustrates the user interface to Errol using BMC 709 as an example.

Information Processing in Errol

In this position paper, we describe the information processing in Errol only briefly; Goel et al. (2020) provides more details. Figure 3 illustrates the flow of information in Errol. The entries in the BMC on the left of the figure are read, tagged with parts of speech terms, and handed off to both a lexical analyzer and a semantic analyzer. Both analyzers generate questions separately and these questions are then combined and sent to the user to the right of Figure 3.

ConceptNet as Errol’s Knowledge Base

Errol uses ConceptNet as its knowledge base (Speer et al. 2017). ConceptNet is an open-source knowledge base that contains information on many everyday terms and represents them as concepts that are related to other concepts. For example, one of the ways a person may describe a doctor is as “someone who works at a hospital”. ConceptNet encodes this information as doctor atLocation hospital, where atLocation is one of many relationships that may result from searching for doctor. Others include capableOf and usedFor, the latter of which says why you may go to a doctor. In addition to existing as a searchable web portal, ConceptNet also provides an API; Errol connects to ConceptNet through this API.

The business model on the input BMC may refer to any of the concept-relation pairs in ConceptNet. Thus, we encoded each concept-relation pair in Errol as a question. These questions make up the “secret sauce” that makes Errol work. Errol takes each noun found in the original BMC and sends an API call to ConceptNet. The JSON that is returned gives us all the concept-relation pairs with which we can generate a question.

Natural Language Analysis

Errol uses NLTK for preliminary processing of the input, namely the entries under VP and CS in the BMC (Bird et al. 2009). In particular, Errol uses NLTK’s part of speech (POS) tagging component to determine the appropriate parts of speech for the code to use. In order to determine what questions to ask, Errol performs lexical and semantic analysis upon its input, as illustrated in Figure 3.

Lexical Analysis

Errol’s lexical analysis begins with a JSON object from the frontend that defines all the inputs and connections between BMC entries. These inputs are then tagged with a part-of-speech tag by NLTK’s POS tagger, and each rule in the code goes through the inputs and tags to determine whether certain questions need to be asked of the user. If a question needs to be asked, it is added
to a master JSON, which is what gets sent to the semantic analysis process.

The lexical system has a list of questions that were manually written and associated with specific triggers. In order to try and make the dialogue between the user and Errol seem a little more natural, each trigger had several questions associated with it, where each question asked the same thing but in several different ways. This way, the user would not necessarily get back the same exact question on every iteration, but the questions would revolve around the same topic from iteration to iteration if necessary.

**Semantic Analysis**

The semantic analysis process is similar to the lexical analysis process. NLTK’s POS tagger is again used to tag the given inputs, but here, the nouns in particular are what are considered. For each noun, a query to ConceptNet is made and each of the relations that ConceptNet has on the noun are used to generate templated questions with the appropriate values plugged in. For example, if *counselor* is used as a search term to ConceptNet, then example relations would be *typeOf* and *capableOf*. Duplicate questions are pruned out and the master JSON is updated before questions are returned to the user.

**Preliminary Results**

Table 1 provides an example of the questions generated by Errol for the business model illustrated by BMC 709 in Figure 2. Errol’s questions in Table 1 are organized by the CS and VP inputs that led to those questions. Goel et al. (2020) provides the results of running Errol on several business models.

For BMC 709 shown in Figure 2, we compared the questions Errol asked of the startup team with questions that human coaches had asked at the I-Corps cohort in Puerto Rico in 2018. Human coaches tend to ask in-depth and thought-provoking questions that induce reflection. They also challenge the ideas behind business models, whether it is the usefulness of what is written on the BMC as it relates to the business model, underlying assumptions and motivations, word choice, or even quantifiers. To facilitate being able to ask these kinds of questions, they ask for more detail but only where things are too vague to understand what the intentions and goals of the startup team are. The main unifying theme is that the questions of human coaches are all backed by a strong understanding of the concept of *business*. While Errol has the same underlying aim as human coaches, its current iteration approaches the teaching challenge from a very different direction, particularly when it comes to the output of the semantic analysis. Unlike how human coaches try to encourage cutting back on things to highlight the more crucial parts of a business model, Errol’s semantic analysis emphasizes questions that prompts the user to expand on ideas. The questions it asks are also more hollow “templated” questions that address the basic “who”, “what”, “where”, “when”, and “how” questions, but they do not necessarily reflect any deep understanding of the BMC as a whole. We note that Errol may even output contradictory questions for the same BMC. Finally, the questions generated by lexical analysis are more similar to expert reviewer questions compared to those generated by semantic analysis.

**Related Work**

Insofar as we know, Errol is the first virtual coach that operates as a Socratic question-asking agent in the domain of entrepreneurship. Here, we briefly situate this work in research on four related topics: computational creativity, knowledge-based AI, intelligent tutoring systems, and question asking. In regard to computational creativity, the open-ended domain of entrepreneurship is similar to some respects to that of design: the problem in both cases is ill-defined, the evaluation criteria are difficult to operationalize, and the evaluation is delayed. Thus, in both design and entrepreneurship, the problem and the solution co-evolve: the problem specification gets revised as the solution takes form. The creative domains of entrepreneurship and design are similar in another respect: both engage the generation of models, business models in entrepreneurship and product models in design (Goel 2013). The major differences between them are that entrepreneurship requires a large investment upfront, the chances of success are low, and the cost of failure can be high. Thus, the margin for error is small.

In reference to knowledge-based AI, Errol makes extensive use of ConceptNet (Speer, Chin and Havasi 2017) that grew out of long-standing efforts to capture commonsense knowledge for interactive applications (Minsky 2004; Lieberman et al. 2004). It is designed to represent general knowledge involved in understanding language and it allows applications to better understand the meanings behind words. The knowledge graph consists of nodes that represent phrases and weighted edges that represent relations between two nodes. Entries in the knowledge graph include pointers to/from many external knowledge bases such as OpenCyc and WordNet. ConceptNet helps Errol understand a learner’s input as it represents the relationships between the phrases. It can also allow Errol to advise on future directions by traversing other edges in the knowledge graph the student might not have explored.

Errol also represents AI research on education as well and can be viewed as an intelligent tutoring system. Graesser, Conley & Olney (2012) and VanLehn (2011) present two overviews of research on AI in
education in general and intelligent tutoring systems in particular. However, unlike many intelligent tutoring systems that address well-defined closed-world problems with a single correct answer, Errol addresses an ill-defined, open-ended problem with no single correct answer known in advance. Further, while most tutoring systems support teacher-guided pedagogy in K-12 education, Errol is intended to support self-directed andragogy for life-long learning.

Question asking is receiving increasing attention in the literature on AI in education, knowledge-based AI, and computational creativity. (Kearsley 1976) and Nielson et al. (2008) provide an early and a recent taxonomy of questions, respectively. Graesser’s (2016) reflections on his AutoTutor project suggest that what questions an AI agent asks is more important than how it asks them; Errol takes a similar stance. In regard to question asking in computational creativity, our work is closest to that on RoboChair (Pollack et al. 2015), which uses templates to ask questions of scientific paper presentations by mimicking a human chair of a scientific conference.

In some ways, Errol may be considered as the opposite of Jill Watson (Goel and Polepeddi 2018), the first virtual teaching assistant. Jill answers learner’s questions; Errol asks questions of learners. Jill replies to very specific questions with precise answers; Errol asks questions in response to ill-defined, vague, and evolving business models on a BMC. Indeed, Errol is more akin to the virtual research assistant VERA (An et al. 2020) than Jill Watson. VERA is an interactive open learning environment for inquiry-based modeling; it helps a learner construct conceptual models of natural phenomena, evaluate the models by simulation, and revise the conceptual and simulation models as needed. In VERA, a learner learns by doing and by reflection. Errol too is an interactive open learning environment for inquiry-based research, and it too helps a learner by doing and by reflection.

**Discussion**

Errol has several limitations compared to human coaches as indicated in the subsection on “Comparison of Errol’s Question Asking with Human Coaches”. Here, we present a brief critique of three aspects of Errol’s information processing that also indicate directions for further work in the near future: the heavy reliance on NLTK’s POS tagging component, the semantic analyzer’s focus on nouns, and the lack of truly iterative question generation.

Errol’s current architecture relies heavily upon the POS tagging component from the NLTK library. The lexical analyzer parses through the POS tags to return questions that the learner would need to reflect on to modify their entries in BMCs. Hence, the accuracy of the questions returned from lexical analysis is heavily correlated to the accuracy of the POS tags. The semantic analyzer also uses the subjects and nouns from the POS tags to query ConceptNet and search relevant terms and relationships. Given that many novice entrepreneurs do not write grammatically correct phrases or sentences, accuracy of the POS tags may be called into doubt. Improving the accuracy of the POS tags can be difficult as NLTK is an external library. Thus, a useful future action would be to develop a method of accurately parsing BMCs that contain grammatically incorrect phrases.

While it has been advantageous for the semantic analyzer to focus initially on nouns, a truly robust question generation system needs to be able to understand every input statement in its entirety. The focus on nouns allowed for the rapid development of initial algorithms for question generation. Yet, other parts of speech can influence what questions should be asked. For example, there may be a difference between asking questions about “small profit margins” versus “large profit margins”. Thus, future iterations of Errol should incorporate techniques like verb and adjective recognition to enhance the set of questions that are produced and asked of the learner.

Truly iterative question generation may be characterized as a technique where a prior set of questions that Errol asks has influence upon a subsequent set of questions. In an iterative system of collaborative argumentative dialogue, if Errol determines that it should pose the same question several times, it likely needs the ability to choose prioritizing that question the next time or to ask fewer questions in total to make that one question stand out. The lack of calling attention to a particularly significant question makes it difficult for the user to see if they are making progress. A future version of Errol will need to address this shortcoming.

**Conclusions**

On the spectrum of creativity, entrepreneurship can be thought of as a new frontier. As entrepreneurship continues to grow in its allure, so too does the desire to learn how to design business models and the need for expert coaching of such designs. We see within this the need for virtual coaches to augment the coaching process. In this paper, we presented Errol as a preliminary startup virtual coach that seeks to simulate the experience of receiving feedback from an expert human coach. Would-be entrepreneurs tend to make the same kinds of mistakes in the various stages of
developing their business models. Human coaches ask questions in the Socratic fashion to help the learners reflect on their mistakes and improve the business models. Errol similarly makes use of typical errors in developing business models to ask questions about a given business model and to help the learner learn by reflection on the feedback. Errol uses both lexical and semantic analysis to generate relevant questions, and in doing so, acts like a sounding board for the would-be entrepreneur. We intend to introduce Errol in a class on entrepreneurship and to assess the students’ use of the virtual coach.

Acknowledgements

Several collaborators contributed to the Errol project over the last five years including Arup Arcalgud, Siddharth Gulati, James Howe, Nikhita Karnati, Jongmin Lee, Aaron Mardis, and Jae Ro. We thank the US NSF I-Corps for its support for this work. KM, the last author on this paper, is a lead instructor with the I-Corps program. We have named our virtual coach after Errol Arkilic, the first I-Corps program director. This research has been supported by internal seed grants from Georgia Tech. This paper is an abridged and much revised version of Goel et al. 2020.

References

Bird, S., Loper, E., & Klein, E. (2009), Natural Language Processing with Python. O’Reilly Media Inc.


Ideation via Critic-Based Exploration of Generator Latent Space

Puneet Jain, Najma Mathema, Jonathan Skaggs and Dan Ventura
Computer Science Department
Brigham Young University
Provo, UT 84602 USA
puneetj@byu.edu, nmathema@byu.edu, jbskaggs12@gmail.com, ventura@cs.byu.edu

Abstract
We present a system for generating, evaluating, and refining logos that can act as a collaborator for creating relevant logo designs. The system combines computational vision and language systems to generate logo design ideations that suitably represent company identity as expressed in a tagline or with keywords. The generative part of the system employs a Generative Adversarial Network (GAN) architecture, while the evaluative part makes use of two vision-based classification models and a language embedding model to assess how well generated logos align with the identity of the company. This process is iterated by using feedback from the evaluation module to guide exploration of the latent space of the GAN. The results may be used as is, or further curated/refined by human designers. For evaluation of the results, two surveys with different sets of participants are conducted. Findings show the utility of feedback-mediated latent space search and that participants rate the system-generated logos above average on creativity and relevance.

Introduction
Generative artificial intelligence (AI) has become quite popular and recently developed models have produced impressive results in domains including visual, musical and linguistic artifacts. Current state-of-the-art approaches for such generative tasks include generative adversarial networks (GANs) (Goodfellow et al. 2014), variational autoencoders (VAEs) (Kingma and Welling 2014) and transformers (Vaswani et al. 2017). All of these approaches leverage deep learning architectures that are trained to minimize loss over a large dataset of artifact examples and generate new artifacts by, effectively, interpolating between abstractions of that training data. Such approaches have been shown to produce very good results, in the sense that the generated artifact is a typical representative of the artifact class (i.e., a person’s face, a song, a story); however, none of these approaches account for the value of the output, in the sense typically considered by computational creativity (CC) researchers [i.e., is it an interesting face?, or a sad song? or an entertaining story? cf. Ritchie (2007)]. This is not surprising, given the generative AI agenda, but the question naturally arises whether such models might be somehow incorporated into CC systems. This paper explores this question by proposing the idea of iterative generation guided by a critic, where both the generator and the critic are deep-learning models that interact via a semantic vector space. To make the ideas concrete, we use them to implement a system for logo creation; we want a system that generates typical logos but also one that generates valuable logos—logos that represent a target company well.

A company logo is something like a visual “meme”, something that incorporates, represents and communicates the company’s identity concisely, visually, and cleverly. The design of a good logo is a time-consuming task requiring not only graphic design skills but also creative thinking and possibly even a bit of serendipity. We present a computational system for logo generation that, while it can be used as a completely autonomous system, may perhaps most appropriately serve as a creative collaborator for human designers, and we give a proof-of-concept of using it in that capacity here—the system iteratively brainstorms and refines a pool of logos for several (fictitious) companies from which we make a final selection.

We are not aware of any extant systems that tackle the task of intentionally creating a logo that serves as a visual representation of identity and/or that communicates concepts. However, there are some systems that tackle similar or related tasks, some of which help inspire our approach here. For example, Özbal, Pighin, and Strapparava developed a system called BrainSUP to support brainstorming creative sentence generation. It was intended to be used collaboratively or as a support tool by a human creator and offers that user/collaborator the ability to constrain the search space in various ways to ensure the sentence communicates desired concepts (2013).

Using a similar framework to BrainSUP, Tomašić, Papa, and Žnidarič used evolutionary computation to build a more autonomous system for creating company slogans. They used eight different evaluation functions to guide the generation, with an aim to produce a slogan reflecting, in some way, the company identity (2015).

On the visual front, Heath, Norton, and Ventura created a computational artist called DARCI which produces images with the intention of communicating one or more linguistic concepts (2014).

More recently, Cunha, Martins, and Machado have explored blending emojis to communicate pairs of linguistic
Figure 1: Model architecture for automatic logo generation. The system is initialized with random $z$-vectors, generates a logo for each using a GAN, evaluates the resulting images based on target keywords, and uses the $z$-vectors associated with the best images as feedback to further explore the GAN’s latent space.

As for logo generation itself, Li, Zhang, and Li apply shape grammars for generating logo designs automatically. The authors encode design knowledge using rule sets and validate the feasibility of their approach via an experiment (2017).

Atarsaikhan, Iwana, and Uchida make use of neural style transfer with clip art and text to generate logos given an input image, introducing a new distance-based loss function to preserve the silhouettes of text and objects (2018).

Sage et al. tackle the problem of logo generation using GANs. Because GANs are data hungry, they scraped the web to create a logo dataset named Large Logo Dataset (LLD) with 600K+ images. They then used this dataset to train a GAN for generating plausible logos, employing synthetic labels obtained through clustering to help stabilize the GAN and prevent mode collapse (2018). Mino and Spanakis built on the work of Sage et al. to condition the network based on color so that a user of this system could have more creative control, ultimately developing a GAN conditioned on 12 different colors (2018).

We build on this recent work, making use of the Mino and Spanakis model as a blackbox generator trained on the Large Logo Dataset (Sage et al. 2018) and drawing a little from the ideas of Atarsaikhan, Iwana, and Uchida, to develop an aesthetic evaluation metric so that the system can autonomously determine the relevance of a generated logo. The result is a system that takes as input the name of the organization and a few keywords that describe the company objective and outputs a set of 100 relevant logos which could be used to inspire ideas (or used as is) for the company’s visual identity.

Contributions of this work include the following:

1. demonstration of iterative guided exploration of the GAN latent space for exploitation/convergent design thinking
2. demonstration of the use of GANs for exploration/divergent design thinking
3. demonstration of computational creativity (CC) system-building using modern, off-the-shelf models
4. autonomous incorporation of linguistic concepts into the visual design
5. collaborative/autonomous creation of intentional visual identity in the form of a logo

**Model Architecture**

The process of designing a logo is expensive and labor intensive. The purpose of our work is to build a creative system that can automate some of the logo generation process, providing suggestions from which a customer might make a final decision or perhaps acting as a collaborator with a human designer.

A diagram of the overall system architecture is shown in Figure 1. At a high-level, the system is composed of two modules: the **generator** and the **critic**. The generator takes as input a $z$-vector which acts as a seed for the logo generation process and is passed as input to the LoGAN model (Mino and Spanakis 2018), which produces a candidate image. The critic takes as input a candidate image and a set of keywords that describe the target identity for the customer company. It then uses two vision-based classifiers—VGG16 (Simonyan and Zisserman 2015) and Pythia (Singh et al. 2018)—that each provide a classification label for the image and a corresponding confidence in that classification. It uses the word embedding model Word2Vec (Mikolov et al. 2013) to vectorize the classification labels and the company keywords and then computes cosine similarity between the keywords and labels to assign a score for the candidate image. Based on this score, the original input $z$-vector is perturbed, and the process repeats. Pseudocode for the two modules is given in Algorithms 1 and 2, and further details are discussed below.
Algorithm 1 CREATELOGO(K)
1: Input: keywords K
2: z∗ ← random normal(0, 1)
3: x∗ ← LoGAN(z∗)
4: ν∗ ← EvaluateLogo(z∗)
5: while not done do
6: z ← z∗ + random normal(0, 0.3)
7: x ← LoGAN(z)
8: ν ← EvaluateLogo(x, K)
9: if ν > ν∗ then
10: x∗ ← x
11: z∗ ← z
12: ν∗ ← ν
13: return x∗

Algorithm 2 EVALUATELOGO(x, K)
1: Input: image x, keywords K
2: ωp, γp ← Pythia(x)
3: ωv, γv ← VGG16(x)
4: δp ← cosinesim(word2vec(ωp), word2vec(K))
5: δv ← cosinesim(word2vec(ωv), word2vec(K))
6: ν ← 1/Z(γpδp + γvδv)
7: return ν

Logo Generation
LoGAN is a Generative Adversarial Network that has learned to generate logos (Mino and Spanakis 2018). It takes as input a 128-dimensional z-vector and returns a generated logo. A z-vector is a point in an arbitrary, condensed feature space. The power of GANs is their ability to learn to organize this latent space such that it is well-behaved, in the sense that each point maps to a reasonable generated artifact and proximal points in z-space map to generated artifacts that are similar (in their original/natural feature space). In the LoGAN model, the latent z-space is therefore an abstraction of the set of possible logos. Each unique z-vector will generate a different logo, and similar z-vectors will generate similar logos. We explore this latent space by initially choosing a random z-vector and then iteratively refining it to perform a local search in its neighborhood. For efficiency and to facilitate diverse exploration, we operate the model on a batch of pools of z-vectors, somewhat reminiscent of specification in evolutionary computation. We initially randomly generate 100 pools of 100 z-vectors (line 2 of Algorithm 1), each of which is used as input to the GAN, resulting in the generation of 100 pools of 100 candidate images (line 3). Each of these images is then passed to the critic for evaluation (line 4).

Critic Evaluation
After a batch of pools of logos is generated, the logos are passed through two image recognition systems (VGG16 and Pythia), which classify the images (one label each, ωv and ωp, respectively) as well as producing a confidence (γv and γp) in that classification (lines 2-3 in Algorithm 2). The labels ωv and ωp are then converted to word embedding vectors (ωv and ωp) using the Word2Vec model (lines 4-5). At the same time, the set of company keywords K are also vectorized using Word2Vec, giving target vectors k1,...,km (also lines 4-5). Next, for each pair (ωv, k1), where * means v or p and 1 ≤ i ≤ m, the cosine similarity is calculated (also lines 4-5):

δs(uv, k) = \sum_i \frac{u_\text{v} \cdot k_i}{\|u_\text{v}\|\|k_i\|}

Finally, an average score ν, weighted by confidences γv and γp, is computed:

ν = \frac{1}{Z}(\gamma_v\delta_v + \gamma_p\delta_p)

where Z is a normalizing constant that accounts for the variable size of K and the confidence values γv and γp (line 6). For each image x the score νx is the critic’s estimate for how well the image communicates the keywords associated with company identity. For each pool j, the image z∗j that has the highest score ν∗j is selected for further exploration, and the vector z∗j associated with that image is used as the location from which to continue exploring the generator’s latent space. Because we used 100 pools, this results in 100 best (so far) images with associated z-vectors.

VGG16 The VGG16 model (Simonyan and Zisserman 2015) is a pre-trained convolutional neural network (CNN) model that won the image classification tasks in the Imagenet Large Scale Visual Recognition Challenge 2014 (ILSVRC2014). The model achieved 92.7% top-5 test accuracy on the ImageNet dataset (Deng et al. 2009), which contains over 14 million images and 1000 output classes. The model is freely available in frameworks like Pytorch or Keras and can be used off-the-shelf for many image classification tasks or it can be partially or completely fine-tuned on additional image sets. For our system, we did not perform additional fine-tuning. The model returns a probability distribution over its vocabulary, and we used this to compute the maximum a posteriori (MAP) estimate and took the corresponding vocabulary word as the output class and that word’s probability under the distribution as the model’s confidence.

Pythia Pythia is a framework created by Facebook Research, built on top of PyTorch, for vision and language research. It is openly available for solving challenges using vision and language datasets (Singh et al. 2018; 2019). Pythia is an excellent tool for recognizing details, answering queries about elements in images, and is different from regular image classifying algorithms, as it provides models that can “read” images. For our system, we used the query “What is shown in the image?” The most confident answer from Pythia was used as output, along with that answer’s confidence.

Word2Vec Word2Vec (Mikolov et al. 2013) is an embedding model for text that transforms words into an n-dimensional vector representation. This vector representation acts as a set of abstract, distributed features that collectively “define” the word. The model is trained on
word co-occurrences in a large text corpus, with the loss designed to organize the space geometrically, such that similar vectors represent words with some semantic relationship and such that different geometric operations on vectors represent semantic operations on words [e.g., the now famous vec("king") - vec("man") + vec("woman") = vec("queen"]). This vector representation is useful because it can be used as input to deep neural networks. As it turns out, this feature representation also allows us to determine mathematically how related words are to each other based on their features, using, for example, the cosine similarity between vectors as a surrogate for the relatedness of the words to which those vectors are mapped.

Exploring z-space

Feedback from the critic is used to intelligently guide the search through the generator latent space by seeding the next round of generation with the \( z \)-vectors representing the current set of 100 best logos (one from each pool, lines 9-12 of Algorithm 1). This is done by computing 100 random perturbations of each \( z \)-vector, resulting in a new batch of 100 pools of 100 candidates (line 6), localized around the best \( z \)-vectors from the previous iteration. These new \( z \)-vectors are again used as input to the GAN to generate a new set of candidate images (line 7) which are again evaluated (line 8). This process is repeated until some stopping criterion is met (line 5). Finally, the last set of best images \( x^*_j \) is returned.

In our experiments, we observed significant improvement in image quality with only a couple of iterations beyond the initial batch generation, suggesting that the process may converge to a set of good suggestions fairly quickly in many cases (see Figure 2).

The returned set of final logos \( x^*_j \) can be further curated by human evaluation based on their appeal and relevance to the company. This final process could be considered analogous to a designer considering a collaborator’s initial ideas or the final selection made by the company’s executive board.

---

**Results**

To test our system, we invented ten fictional companies for which to create logos. We used fictional companies to avoid potential familiarity bias in the system (via data used for model training), during logo post-selection, and in survey respondents during external evaluation. Table 1 presents the company names and the related keywords used for generating the logos for each company. After the system produced its final batch of logo suggestions, we manually post-selected the logo we felt worked best given the company name and keywords, simulating the final say the company administration would have in the process. Figures 3-6 present the most (subjectively) appealing logos generated by the system for the respective companies.

The logo for Star Creme Coffee looks like a coffee cup viewed from above, with the suggestion of a ‘C’ inside. The logo for Juice Juice is a simple stroke ‘J’ on a bright yellow background. Both are (subjectively) quite appropriate (see Figure 3). Figures 4 and 5 show other logos created, some more successful than others: the Bakery logo is suggestive of rolls or loaves of bread; the Sporting Goods logo looks like a ball (perhaps tennis or basketball); the Papa Pizza logo is perhaps reminiscent of a pizza box; the other three look like logos of some kind, but it is more difficult to justify their relevance to the target companies, though the Lu Lobster logo’s color is suggestive of lobster, the Food Barn logo could be a (grain) scoop and the MH Clothing logo could be initials.

Perhaps most striking are the logos generated for the companies Rocky Mountain Bikes and T-Sprouts Babywear, shown in Figure 6. The former looks like both a bike rider and a mountain range with the sun, with text at the bottom; the latter looks like a parent holding the hand of a toddler within a heart frame.

Some other interesting logos which give us more insight
In order to perform an external evaluation of the system’s effectiveness, we designed surveys to address the following questions:

1. Does exploring the GAN’s latent space using feedback from the evaluation module add value over traditional GAN generation?
2. Are the system-generated images good logos for their respective companies?

**Efficacy of Latent Space Search**

In order to test whether feedback-guided search of the latent space aided in the design process, we used the following experimental design. For each of the ten companies:

1. The system is run to generate a set $A$ of $n$ logos.
2. The set $A$ goes through a final curation process by which the logo $a$ deemed to best meet the company’s design requirements is chosen by hand (in essence, we simulated acting as the design team for the company, working with a collaborative CC system). The generated logos will be referred to as ReleLogos [short for “Rele(vant) Logos”] from here on.
3. The GAN of Mino and Spanakis is run to generate a set $B$ of $5n$ logos.
4. The set $B$ goes through a final curation process by which the logo $b$ deemed to best meet the company’s design requirements is chosen by hand (again, simulating a design team for the company, with a different, less-collaborative, more tool-like system). These are called Random Logos.
5. A survey is conducted in which we ask, for each company, which logo, $a$ or $b$ (ReleLogo or Random Logo), is preferred. To avoid bias, we randomized the pair ordering for each question (see Figure 8).

68 participants, from geographic locations in India, Nepal, the UK and the USA, responded to the survey, giving us 680 responses (68 for each of the 10 companies). Table 2 shows the total number of votes obtained for each of the companies for the ReleLogos and the Random Logos. A Paired $t$-test shows that the difference in average votes obtained for the ReleLogos and Random logos is big enough to be statistically significant ($p = 0.04011$).

The final row of the table shows that ReleLogo logos received nearly twice as many votes overall, and for 8/10 companies, our system’s logos were preferred by wide margins. Because both types of logos were post-selected by human “designers”, the fact that ReleLogos were chosen nearly 2:1 over the more random logos provides strong evidence that the feedback-based latent space search is providing significant value in the design process.

As an additional bit of anecdotal evidence for this claim, we note that in an informal survey of generated sets $A$ and $B$ for each company, the preferences were nearly identical to those in the formal survey.
B, it was observed that ≈10% of the logos generated by our system (set A) appeared to be relevant to company identity, whereas in the more randomly (LoGAN) generated dataset (set B), less than 1% were relevant in any way, suggesting a potentially 10× improvement in efficiency of the design process.

Logo Quality

In order to evaluate how well the logos our system generates might be received as creative artifacts, we designed a second survey inspired by Jordanous’s evaluation guidelines for computational creativity (2011). We identified four characteristics that might describe the goals for logo design: visual coolness, relevance to company identity, intelligence of design, and perceived creativity. We chose the seven logos we felt best demonstrate the capability of our system, and asked respondents to rate each of them for the four characteristics. Because these were not relative to any baseline, we settled on a 3-item Likert scale for evaluating each characteristic: yes, maybe, no (see Figure 9).

To avoid bias, the second survey involved new participants (20 total), and we targeted people with experience in computational creativity research.

The results are depicted in Figures 10-13. Most notably, the Rocky Mountain Bikes logo was clearly a success, with only a single no vote across all four criteria. In addition, 5/7 logos were pretty clearly considered at least somewhat cool looking; 6/7 were considered at least somewhat relevant to company identity; 4/7 were considered at least somewhat intelligent design ideas; and 5/7 were considered at least somewhat creative.

Table 3 is an agglomerated view of various averages across the 20 participants (Likert values were converted to numerical values as no: 0, maybe: 1, yes: 2). The first four rows show average (across companies) response rates for each of the characteristics—while there are clear wins for each of these at the company level, some less successful designs result in overall ambivalent averages; still, on average, the system does not fail for any of the characteristics. The next seven rows show average (across characteristics) response rates for each of the companies—here there are some clear standouts, both positive (Star Creme Coffee, T-Sprouts Babywear and Rocky Mountain Bikes) and negative (Lu Lobster and Papa Pizza). The last row is an overall average across both companies and characteristics—a somewhat encouraging maybe.

Discussion

The GAN-based approach of Sage et al. provides an interesting model and an end-to-end pipeline for logo generation. However, because it has no mechanism for conditioning the output, nor for the system to self-evaluate the resulting logos, it is of limited use for facilitating the process of logo design and selection for any particular company—the vast majority of the output logos will always be irrelevant to any specific designer or company, even though they will look generally like logos. Mino and Spanaklis improve on this by allowing some conditioning (the designer can have some color control), but their improvements still result in far too many irrelevant outputs.

Building on their work, we have shown how a GAN-based
Each company logo design is rated on four characteristics: intelligence, coolness, relevance, and creativity.

The fact that the search for quality logo images is conducted in the GAN’s latent space (rather than in the much more complex raw pixel space) both allows the process to be more efficient and to incorporate linguistic information (in the form of vectorized embeddings).

**Conclusions and Future Work**

This paper presents a novel methodology for generating and evaluating logos for a specific company or a brand. The benefit of this approach is the ability to intelligently search the latent space of the generator using feedback from visuo-linguistic evaluation, and the general approach should be immediately applicable to many other visual creation tasks. While our methods currently most naturally apply to visual tasks, with additional work it is possible that they may be further generalized, either by generalizing GAN models beyond visual generative tasks or by using other types of generative models (e.g., transformers) in their place.

While the system can be used autonomously, with full creative control, we envision it more as a collaborator, and, for now, best results are obtained with some human post-selection of the final system output.

Two different external evaluations verified that

1. intelligent search of the generator’s latent space provides value over random generation, even given hu-
2. the resulting logos are perceived as sometimes cool looking, reasonably intelligent ideas, and generally both relevant and creative

While these survey results are positive, they should be considered as encouraging preliminary indications that demand further validation with more rigorous and larger scale evaluation instruments.

The model does not yet deal well with text on the images. Pythia was queried for “What is written on the image?”, but it did not yield good results. For future work, it would be better to implement Optical Character Recognition (OCR) for text-based logos. Logos with better designs often have better to implement Optical Character Recognition (OCR) and image resolution would also likely improve final results. For future work, it would be more problematic, but other vision systems could also be considered. If such a loss function can be realized, this opens up the possibility of backpropagating loss through the vision system and then through the generator back to the initial \( z \)-vector input. In this way, the latent space could be searched using gradient descent rather than using the random perturbations we use now, likely resulting in significant improvement in both system run times and quality of output. The vision/classification and generating systems could be trained at the same time, or, they could still be used as off-the-shelf modules as has been done here.

### References


Abstract
Portmanteaus are a type of neologism combining two source words, for example brunch (from breakfast and lunch), and are popular for naming all kinds of phenomena. While coming up with suitable portmanteaus is a difficult creative endeavor, several portmanteau generators already exist for assistance in this process. When using these systems, it is often hard to find out which of the generated portmanteau is likely to be the best, and consequently also hard to automatically compare the quality of different portmanteau generators. In this paper, we create a model that can rank portmanteaus for two given source words, which thus aims to help find the best portmanteau to help further improve portmanteau generators. Our model first uses XGBoost trained on unlabeled generated outputs and existing portmanteaus to learn to rank portmanteaus and shows that this already greatly improves the performance of the initial generator. By ranking outputs of a state-of-the-art generator and a new simple portmanteau generator, we show by validating its quality in a human evaluation that the ranker can help visually identify the better generator, thus providing an alternative to only calculating real portmanteau generation frequency. Additionally, we find that this first model performs almost as well as a model trained on more fine-grained human-labeled portmanteaus. This indicates that just using generated and real portmanteaus is enough to create a ranker that can in turn improve the quality of the initial generator, and could additionally be of use in comparing different portmanteau generators.

Introduction
A portmanteau is a type of neologism mixing two words based on their pronunciation and meaning, where the resulting word is not simply concatenating the two words into a normal compound word, for example brunch (which mixes breakfast and lunch). This wordplay is popular in all kinds of domains, such as naming new objects (e.g. jeggings for jeans leggings), pop culture (e.g. for relationship names like Brangelina for Brad Pitt and Angelina Jolie), animals (e.g. crossbreeds like liger for the child of a tiger and lion) and company names (e.g. Netflix for internet ficks). There already exist several portmanteau generators that use heuristics or neural networks for estimating the quality of portmanteaus in their generation process (Smith, Hintze, and Ventura 2014; Deri and Knight 2015; Gangal et al. 2017; Simon 2018). In this study, we focus on the task of portmanteau quality estimation, which can help further enhance the generation quality of any other portmanteau generator by sorting the outputs on their perceived quality. To achieve this, we create a portmanteau evaluator for further ranking the outputted portmanteaus. Additionally, this technique further allows comparing different portmanteau generators. More specifically, we implement a model ranking the quality of a merge of two input source words, and show that this model can be effectively trained using only generated and existing real portmanteaus.

Background
Portmanteau Generators
Nevehoh is a collaborative, rule-based system for generating portmanteaus that use synonyms and hyponyms of the input source words to enrich the search space (Smith, Hintze, and Ventura 2014). The user can decide which portmanteau factors are most important. It searches for possible letter-level overlap, thus limiting its possibility to generate certain portmanteaus without letter overlap (e.g. brunch).

Frenemy is a more data-driven approach for generating portmanteaus (Deri and Knight 2015). It trained a multi-tape finite-state transducer to map the source words to the portmanteau using existing portmanteaus from Wikipedia and Wiktionary. By mixing the two given source words, it generates the real portmanteau 45% of the time.

Charmanteau extends the dataset used in Frenemy with Urban Dictionary and BCU Neologism List dataset (Gangal et al. 2017). This model does not use explicit grapheme to phoneme conversion but uses character embeddings to train a noisy channel model, maximizing the probability of generating the correct portmanteau given the source words. This is implemented as a neural sequence-to-sequence model using LSTMs and attention. When generating portmanteaus for two given source words, its implementation only returns the top 5 portmanteaus.

Entendepreneur uses FastText word embeddings to find the best related source words for creating a portmanteau (Simon 2018). The focus of this system is more on finding words with good overlap, and thus much less on finding the ideal way of merging two words. As such, many existing
portmanteaus can not be generated with this system.

**Learning to Rank**

In a learning to rank problem, the task is to predict the order of a given set of elements. A popular system for doing so is XGBoost trees (Chen and Guestrin 2016). This algorithm uses a form of tree boosting, which iteratively extends simple decision trees with other simple decision trees to minimize the quadratic error of the model. There are three types of ranking models, namely those that predict scores for each element, compare elements pairwise, or list-wise. While the last one usually gets the best performance, they are the most difficult to model and train. In this paper, we thus focus on pairwise comparison.

**Ranking Portmanteaus**

To rank portmanteaus, we employ the following features, inspired by the aforementioned portmanteau generators.

- **Word structure**: proportion the source words present in the portmanteau by calculating the length of overlap between the cut-off parts of the source words and the actual portmanteau (Smith, Hintze, and Ventura 2014).
- **Source word fraction**: fraction of the length of the longest part of the source words present in the portmanteau.
- **Fraction of syllables**: number of syllables of the portmanteau divided by the number of syllables of the source words, as well as for each source word separately.
- **Memorability**: meaningful character ratio (Schiavoni et al. 2014), which is the ratio of meaningful subsequent characters that create a word present in WordNet (Miller 1995).
- **Pronounceability**: weighted frequency of letter n-grams ($n \in [2, 4]$) in the Wordlist Corpus from NLTK (Bird, Klein, and Loper 2009), since popular longer n-grams serve as proxy for pronounceable words.
- **Length of portmanteau** the absolute length of the portmanteau, and also features for the difference in length between the portmanteau and both source words separately.

These features are used to train an XGBoost model that learns to rank real portmanteaus higher than generated ones. This is done by assigning a weight of 1 to the real portmanteau, and a weight of 0 to all portmanteaus generated by the portmanteau generator for the same source words. While this assumes generated portmanteaus are worse than existing portmanteaus, it might still result in a model that can distinguish portmanteaus on their quality, as this assumption is usually true (as confirmed by our evaluation). One downside to using this way of ranking is that it might assign the same rank to multiple elements.

**Simple Portmanteau Generator**

We want to evaluate whether the ranker can differentiate between different portmanteau generators. However, due to the nature of the ranker and its dataset, we only want to compare their word combiner algorithm, rather than their synonym exploration capabilities. As such, comparing to generators like Entendrepreneur and Nehovah (which focus on finding good related words) would unfairly disadvantage them. While Charmanteau and Frenemy are thus the only documented models we could meaningfully compare, we were only able to gain access to the code of the former. We thus created a simple portmanteau generation algorithm, that while generating decent portmanteaus, should perform measurably worse than Charmanteau due to its lack of portmanteau quality knowledge.

We use three simple mechanisms simultaneously in the simple portmanteau generator, namely splitting on hyphenation, finding mutual letters and random splits. First, the source words are split using Pyphen 1. If source words have letters in common, the naive generator adds combinations of these subwords split on the common letter, except if the common letter is the first letter of the first source word or the last letter of the last source word. After these generation methods, the algorithm adds five additional portmanteaus by splitting the source words into random possible substrings larger than 1, and merging randomly to create the remaining candidate portmanteaus.

**Data**

**Real Portmanteau Dataset Extension** We extended the dataset from Charmanteau with the more recently added portmanteaus from Wikipedia. We then filtered out normal compound words, portmanteaus based on proper names (like Jedward), fandom names (like Cumberfan) and chemical compounds (like glyoxime).

**Negatives Generation** For the source words of each real portmanteau from the Wikipedia dataset, we used Charmanteau to generate five portmanteaus, which we use as negatives for training the ranking model.

**Human-Annotated Labels** We generated a portmanteau dataset and annotated this to evaluate the human-perceived quality of portmanteaus. We did this by first training an XGBoost ranking model on the aforementioned dataset. Then, for 700 real portmanteaus, we generated portmanteaus using Charmanteau and our simple generator, and took the top 4 of each list according to our ranker, and added the first three that were not the real portmanteau, resulting in 700 portmanteau groups of at most 7 possible portmanteaus. Each generator thus provided 2100 portmanteaus, of which there was an overlap of 331 that both generated. The annotators were 10 non-experts, recruited by sending a link to the annotation platform to willing friends of the first author. Each human annotator could annotate as many or as few portmanteau sets as they wanted. Human annotators were allowed to annotate these portmanteaus with labels denoting first, second or third place, and also annotate portmanteaus as “very bad” for portmanteaus that were fundamentally worse (e.g. due to being hard to pronounce) than all others. In case a human perceives two portmanteaus to be of equal quality, the...
labels could be used multiple times within the same portmanteau group (e.g. shared first place).

Evaluation

In the evaluation, we aim to answer the following questions about our portmanteau ranker:

Q1 Is the ranker better at identifying existing portmanteaus than previous approaches?
Q2 Can the ranker improve the quality of the outputs of existing portmanteau generators?
Q3 Can the ranker evaluate the quality of two given portmanteau generators, and help identify the better one?
Q4 How well does the ranker trained on generated negative portmanteaus approximate the human-perceived quality?
Q5 Is just knowing the real or best portmanteau enough for improving generation quality? In other words, how fine-grained does the ranker training dataset need to be?

Identifying Real Portmanteaus

Due to multiple elements possibly being assigned the same rank, we break the ties in favor of the real portmanteau and in favor of the generated portmanteau to find the best and worst-case performance. We compare this with Charmanteau in Table 1. Note that Charmanteau is only able to generate 55% of the portmanteaus of our test set as one of the five candidates it generates and that we thus only use this part of the test set to more fairly compare the ranking capabilities of Charmanteau and our ranker. In the worst case, 58.63% of all portmanteau groups the right one was identified in the first position, and in the best case 72.57%. This means that either way, the ranker is better at identifying the best portmanteau given the top 5 possibilities generated by Charmanteau. Note that due to the nature of portmanteau quality in relation to its real daily use, false positives might actually still be good portmanteaus too (e.g. sendex instead of sensex for sensitive + index, and plebevision instead of plebvision for pleb + television). To answer Q1: our ranking model does indeed perform better than the state-of-the-art in identifying the real portmanteau.

Several portmanteau research projects used the real portmanteau metric to measure a generator’s quality. Given that we used only Charmanteau generated portmanteaus to augment the dataset, adding the ranker to filter the generator’s output would help it achieve higher scores on this quality metric on average, thus positively answering Q2.

Comparing Portmanteau Generators

We hypothesize that this ranking model could potentially be used as a form of automatic quality evaluation. Portmanteau generators are often evaluated in terms of how many times the real portmanteau can be reconstructed (Gangal et al. 2017; Deri and Knight 2015). A ranking model could automatically rank generated portmanteaus from different sources. Assuming the ranker does a good job estimating relative quality (which it does according to Table 1), the higher a generator’s outputs are ranked compared to the other, the better one would expect it to be. This would then create a less accurate metric than full human evaluation for quality, but a much easier metric for comparing portmanteau generators to each other. By visually plotting how high generated portmanteaus are ranked, one can thus see which rankings are more populated by which algorithms, regardless of how many outputs are generated by each generator. This thus provides a more detailed way than only comparing real portmanteau generation frequency.

We used our simple portmanteau generator and Charmanteau to generate portmanteaus for the source words of the test dataset and made our ranker rank all given portmanteaus. The algorithm ranked both lists separately, then merged them in a similar way as the merge sort step by comparing the first elements of each queue. If there was a tie between the highest of the two lists, then it counted as a tie, which can be ignored in the visualization. If one is better than the other, it polls the highest from the two and adds this as the current rank for this generator’s element. Counting how often a generator generates elements at a certain rank, then visually shows how well their generated portmanteaus

<table>
<thead>
<tr>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charmanteau</td>
<td>45.70%</td>
<td>20.97%</td>
<td>15.01%</td>
<td>9.71%</td>
</tr>
<tr>
<td>Ranker (worst case)</td>
<td>58.63%</td>
<td>17.26%</td>
<td>10.62%</td>
<td>7.30%</td>
</tr>
<tr>
<td>Ranker (best case)</td>
<td>72.57%</td>
<td>12.83%</td>
<td>7.74%</td>
<td>4.65%</td>
</tr>
</tbody>
</table>

Table 1: Comparing how often the true portmanteau is ranked as the best portmanteau given four other portmanteaus generated by Charmanteau using the 59.38% of the test set where Charmanteau is actually able to generate the real portmanteau. The worst and best case reflect the tie-breaking mechanism in the ranker.
perform compared to the other. The results in Figure 1 show what we expected, namely that Charmanteau performs better than the simple algorithm. The chart makes this clear by showing that Charmanteau peaks at the 3rd rank, and the simple generator at 1st, but also at the 9th position, and the fact that most of the mass of the simpler algorithm is ranked lower than that of Charmanteau. This is because the simple generator generates much more candidates (at least five per input source words, while Charmanteau always generates exactly five), and thus on average worse ones while also being able to generate some good portmanteaus that Charmanteau can not or did not. This is in line with Figure 1, where we can see that according to the human evaluation, Charmanteau indeed generates 59.89% of the highest-ranking portmanteaus, while the simple generator only 54.40%, and also much more of the very bad and fewer higher-ranking ones. While we used ranked versions of both generators in the human evaluation, the fact that the ranker and the human evaluation both confirm that the simple generator performs worse (as it is designed to do), helps answer Q3. This in combination with Figure 2 answers Q3: yes, the ranker can help identify and visualize the better performance of Charmanteau compared to the simple generator. This fact is a promising result for the automatic evaluation of generators, given the easy-to-create ranker training dataset.

Figure 2: Human evaluation annotations for the portmanteau sources, with at most seven portmanteau per group of source words (existing portmanteau plus at most three of each source, with potentially fewer due to both generating the same portmanteau).

**Dataset Granularity & Human Evaluation**

As mentioned in the Data section, we use the human annotations to create datasets for training a portmanteau ranker to evaluate the best dataset granularity and the need for human-annotated labels. Linking the resulting labels with the source of the portmanteau (Figure 2), we can see that while most of the highest-ranked portmanteaus were the real portmanteau, the generators also came up with high-ranking portmanteaus that weren't the real portmanteau. An example of a generated portmanteau that was ranked first, is *brocket* for *broken + bracket* (whereas the real one is *broket*. An example of a portmanteau labeled “very bad” for this pair was *bket*.

The human annotations were used to create three new datasets. One dataset is similar to the first ranker using only labels reflecting which portmanteau is used in reality (called “Only real”), one with only the highest-ranking ones weighted as 1, and all others as 0 (named “Only highest”), and the third dataset (called “All annotations”) using weight 3 for first place, 2 for second place, 1 for third place, 0 for unlabeled and -2 for the “very bad” label. This extra information about many possible pairs would in theory help ranking portmanteaus in a more detailed way. We trained the same XGBoost ranker on our three new datasets. Table 2 shows that the ranker trained on only the real portmanteaus can still predict which one is performing best with 1.44% less accuracy than the one trained on the truly highest-ranking portmanteaus. This answers Q4, as just using real portmanteaus seems to work about as well as annotated high ranking portmanteaus. This can be partially explained by the fact that properties of real portmanteaus are closely correlated to the first place ranking portmanteaus, and that 60.03% of the real portmanteaus are also labeled best (Figure 2), thus validating our assumption for generating negative examples.

To answer Q5, we compared the last two rankers, which use a different label coarseness. We can see that adding more fine-grained annotations does not significantly improve the performance in predicting the best portmanteaus. Additionally, the higher number of 4th and lower-ranking real portmanteaus could indicate light degradation of the model for predicting the best portmanteaus.

<table>
<thead>
<tr>
<th></th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>≥5th</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only real</td>
<td>42.58%</td>
<td>19.14%</td>
<td>12.92%</td>
<td>11.96%</td>
<td>13.40%</td>
</tr>
<tr>
<td>Only highest</td>
<td>44.02%</td>
<td>20.10%</td>
<td>12.44%</td>
<td>10.53%</td>
<td>12.92%</td>
</tr>
<tr>
<td>All annotations</td>
<td>44.98%</td>
<td>18.66%</td>
<td>9.57%</td>
<td>11.96%</td>
<td>14.83%</td>
</tr>
</tbody>
</table>

Table 2: Comparing how often the highest-ranking portmanteau according to human evaluation is ranked as the best portmanteau out of the seven possibilities for each portmanteau group. Ties were broken not in favor of the actual highest-ranking (worst case rank).

**Code and Data**

We released our code and data on [https://github.com/larapollet/portmanteau-ranker](https://github.com/larapollet/portmanteau-ranker).

**Future Work**

It would be interesting to validate if this approach also works for improving other types of generation, such as other types of wordplay e.g. acronyms and anagrams, or even non-textual domains. For example, by taking existing, funny anagrams, generating some other ones with a simple generator, and learning to rank anagrams, which in turn improves the average quality of the outputs the simple generator by...
only outputting the best few. Another interesting further extension would be to allow for ranking portmanteaus based on semantically related source words and thus have a model ranking more diverse portmanteaus, similar to the Frenemy and Entrendrepreneur generators.

Conclusion
We created a model for automatically ranking portmanteaus. We showed that using only real portmanteaus, and generating other portmanteaus with a particular generator, can be used as a training dataset for a ranker that in turn can help improve the average quality of the outputs of that generator. This relies on the assumption that real portmanteaus are generally better than other possible portmanteaus, which we confirmed in our human evaluation. We also found that using human-annotated portmanteaus only slightly improved the quality compared to using only using labels reflecting whether or not the portmanteaus were used in real life. We also found that the largest increase in performance mostly came from only using the label for the best portmanteaus. Both these findings give rise to an optimistic view about the ease of data collection when replicating this model to improve the output quality of other types of text generators.

Acknowledgments
We would like to thank the volunteers for labeling the portmanteaus. Thomas Winters is a fellow of the Research Foundation-Flanders (FWO-Vlaanderen, 11C7720N). Pieter Delobelle was supported by the Research Foundation - Flanders (FWO) under EOS No. 30992574 (VeriLearn) and also received funding from the Flemish Government under the “Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen” programme.

References
SOVIA: Sonification of Visual Interactive Art

Lauryn Gayhardt and Margareta Ackerman
Department of Computer Science & Engineering
Santa Clara University
{lgayhardt,mackerman}@scu.edu

Abstract
This paper presents SOVIA, an interactive system that endows Claude Monet's art with responsive auditory experiences. SOVIA uses computer vision trained on Monet’s artwork to take the user “into the painting.” When the user interacts with a digital version of Monet’s landscapes, their mouse positions are mapped to sounds that artistically represent the objects that the user is currently exploring in the art. These interactive musical journeys have the potential to make classical art more captivating for modern audiences.

Introduction
In recent years, entertainment has been becoming progressively more interactive - from social media, art, and education, there is an effort to engage users beyond mere consumption. This opens up the challenge of how to endow classical art forms with new layers of interactivity to engage modern audiences, while retaining the essence of the original art.

Visual art uses color, light, texture, and stroke techniques to convey the mood, tone, and meaning of the artwork. Every layer of information aids in expressing the artist’s intent. Adding music and sound to a painting can assist in creating more depth, strengthen existing themes, and convert a consumption-based experience to an interactive one.

In this paper, we propose an interactive method for deepening engagement with visual art. We introduce SOVIA (Sonification of Visual Interactive Art), an interactive system that adds an auditory dimension to still art. Our initial version of SOVIA endows Claude Monet’s landscape paintings with soft music and nature sounds, which respond to user’s mouse position. The aim to bring the user “into the art,” letting them experience a self-directed musical journey into Monet’s landscapes (see https://github.com/lgayhardt/SOVIA for a demo).

Monet was a French impressionist artist (1840-1926) and the first to paint outside the studio. He aimed to capture “what is seen rather than what is known” (Seitz 2021). SOVIA’s sounds added to his paintings amplify the idea that the viewer is experiencing a slice of life.

Instead of trying to accurately reproduce the scene before him in detail, Monet aimed to record “on the spot the impression that relaxed, momentary vision might receive—what is seen rather than what is known, with all its vitality and movement” (Seitz 2021). To reflect this balance of the literal and metaphorical, SOVIA adds an auditory dimension that intermixes real sounds with musical elements.

Using machine learning enabled object detection, SOVIA recognizes objects in the paintings, which are subsequently mapped to sounds. When the user glides their mouse over a hill they will hear sounds of herding bells through the background music. If the user’s mouse wonders over flowers, chimes will play, similar to what one may hear in a garden as a soft wind floats by. This mixture of music with realistic and associated sounds creates an experience that mimics realistic elements in the art, while reflecting the gentle artistic reinterpretation of those objects through sound.

The interactive process offered through SOVIA places most of the effort on the machine agent, while giving the user a simple and enjoyable experience that deepens engagement with Monet’s landscapes. We hope that the process proposed here will inspire more researcher into how creative machine agents can be used to enliven classical art forms.

Method
SOVIA is written in Python, using Microsoft Azure’s Custom Computer Vision Service for object detection and Pygame to handle tracking the mouse movements and playing the sounds. When a user’s mouse enters the coordinates of a detected object it will select an associated sound and increase the volume of the sound to an audible level.

Computer Vision
We manually created the training data for the model. The training data consisted of 369 various sized Monet landscape paintings with the tags: flowers, snow, structure (building), grass, mountain, water, sky, and boat. The training set includes 55 tags for flowers, 53 for snow, 158 for building, 161 for grass, 57 for mountains, 212 for water, 57 for boat, and 318 for sky. The model was trained for five hours.

To evaluate the model’s performance, we consider precision, which is the likelihood that a tag predicted by the model is correct; recall, the probability that the model found all the objects in a given image, and mean average precision (mAP). Our model has an 80% probability threshold with 62.2% precision, 71.2% recall and 64.8% mAP.
Figure 1: **A visualization of the internal workings of SOVIA.** The bounding boxes detected using computer vision are labeled and mapped to a set of sounds, one of which is played when the cursor enters the box. As the user explores the painting through mouse movements, sounds corresponding to the different bounding boxes are intermixed with the background music. Bounding boxes are not shown to the user. The sky boxes are mapped to sounds of a strong gust of wind, the tree boxes are mapped to bird sounds in Cát Tien National Park Vietnam, while the flower box is mapped to sounds of chimes. The position of the cursor dictates the active box whose sound is played, visually represented here through highlighting in this image. The painting is of Poppy Field at Giverny (1890) by Claude Monet. Photo Credit: WikiArt
The custom vision service uses normalized coordinates with left, top, width, and height to detect objects. In this project, normalized coordinates are converted into pixel coordinates to compare the mouse location in Pygame. Each tag has corresponding sounds associated with it, some tags have one sound while others have many. When there is more than one sound associated with a tag, the sound to be played is randomly selected. All tags are mapped to a Pygame sound channel, which starts at a volume of 0.

When the mouse location intercepts the boundary for a detected tagged box, it increases the volume for the sound channel for that tag until it reaches a clearly audible volume which it will remain at until the mouse is moved outside the boundary box. When the mouse is moved outside the box, the volume starts to decrease until it is 0 again. Increasing and decreasing the volume allows for sounds to continuously play, so when the mouse intercepts the boundry for a box, the sound does not start over again.

If there is more than one of a certain type of tag detected, the same sound is still played for that tag. For example, if an image has two tags of trees, one of the many tree sounds will be selected to be played for all tree tags. When the mouse intercepts one of the tag box’s, the volume for that sound will be increased.

Sounds and music are royally free and were obtained from Zapsplat, Freesound, Mixkit, and Avosound. Please see Figure 3 for a list of object-sound mappings used in SOVIA. Some tag sounds were chosen for their literal representation of the tagged feature while others are more abstract. For example, Chimes were used for the flower tag to create the feeling that one is walking into a garden and hears chimes blowing in the wind. Chatting in Polish is used in the building to give the sense of a lively conversation taking place within the house. Goats are heard for the mountain tag because it is one of the many animal sound that one may hear on a mountain top.

Related Work

Related work has considered the conversion of art to music and vice versa, often focusing on creating novel works rather than enhancing existing art works. Other related work offered co-creative experiences which allow the user to concurrently create new music and art. We share several examples here.

MetaSynth (U & I Software 2021) lets users create music from images. It allows for pixels to be drawn or imported onto a digital canvas, and uses the RGB color data from pixels to affect and produce sound. It generates audio using the brightness of the pixels to control to amplitude, pitch bases on the pixel’s Y-axis position, and the red and green color components to pan the audio to the left or right speaker channel (Pitman 2009). Metasynth differs from SOVIA in that it generates audio at the pixel level of an image vs detecting objects within an image that map to different sounds to be...

<table>
<thead>
<tr>
<th>Tag</th>
<th>#</th>
<th>Sounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trees</td>
<td>10</td>
<td>Forest sounds with birds</td>
</tr>
<tr>
<td>Sky</td>
<td>5</td>
<td>Wind sounds</td>
</tr>
<tr>
<td>Flower</td>
<td>1</td>
<td>Chimes</td>
</tr>
<tr>
<td>Grass</td>
<td>1</td>
<td>Cicadas with distant birds</td>
</tr>
<tr>
<td>Water</td>
<td>7</td>
<td>Sea, stream and underwater sounds</td>
</tr>
<tr>
<td>Mountain</td>
<td>2</td>
<td>Goats and herding bells</td>
</tr>
<tr>
<td>Snow</td>
<td>3</td>
<td>Footsteps in the snow</td>
</tr>
<tr>
<td>Boat</td>
<td>4</td>
<td>Boat swaying in water, creaking wood and paddling</td>
</tr>
<tr>
<td>Building</td>
<td>1</td>
<td>People chatting in Polish</td>
</tr>
</tbody>
</table>

Figure 2: The Castle in Antibes (1888) by Claude Monet with bounding boxes on water, sky, and building objects. Photo Credit: WikiArt

Figure 4: Our computer vision model with 80% probability threshold performance metrics. Photo Credit: Microsoft’s Custom Vision Portal
played.

PhotoSounder (Rouzic 2020) is a graphical and audio editing software and synthesizer that allows the user to make music from images. It is similar to MetaSynth, but gives the user greater control over sound mappings and runs on both Windows and Mac.

PixelSynth (Jack 2016) is a browser based synthesizer inspired by the analog ANS Synthesizer that creates sounds from images and drawings. It uses the grayscale version of an image and turns it into a sine wave by having the white in the picture represent a note that is on, transparency representing velocity of the note and location for pitch (Arblaster 2016).

A system by Joana Teixeira and H. Sofia Pinto (Teixeira and Pinto 2017) takes an image as an input and generates music by relating visual features to musical ones. Conversely, a system by Luís Aleixo, H. Sofia Pinto, and Nuno Correia (Aleixo, Pinto, and Correia 2021) taking in music which it uses to generate an abstract image.

SOVIA differs from the systems above because it is not transforming one art form into another, or offering a co-creative experience that creates novel art and music, but rather amplifying a given artwork by adding a musical interactive experience.

Related Digital Art

Digital artists have also experimented with interactive integration of sounds and visuals through online installations. For example, Joe Hamilton’s Indirect Flight is an interactive web art that displays a layered collage of landscape images set to a realistic soundscape of wind aeroplanes and other urban noises by J.G Biberkopf. “As you pan across the terrain like Google Maps the layers move at different speeds giving the illusion of depth, constantly changing what is hidden and exposed.” (Hamilton 2015).

Rafael Rozendaal’s Sunrise/Sunset allows users to interact with circles that represent the rising and setting of the sun in New York City on the Whitney Museum of American Art website creating a piece of abstract art. “As visitors to whitney.org move the cursor over the black or white circle obscuring the web site, they cast spinning shadows or light over the page that obscure or expose its content” (Whitney Museum 2017).

Conclusions and Future Work

Creative machine agents can add interaction to traditionally consumption-only visual art experiences. The interaction can result in a more immersive experience and perhaps invite viewers to more deeply engage in meaning formation.

SOVIA adds an auditory component to traditional visual art, particular the works of Monet. Adding acoustics that blend realistic and symbolically-related sounds with soft music pairs well with Monet’s style - inviting viewers to a self-directing sonified experience of Monet’s landscapes.

Future work will carry out user studies comparing the consumption of artwork with and without the interactive musical dimension added by SOVIA. We would also like to expand SOVIA to other artists, as well as allow users to upload their own images which the system will endow with an interactive musical dimension.

References


Creating new Program Proofs by Combining Abductive and Deductive Reasoning

Kuruvilla George Aiyankovil, Diarmuid P. O’Donoghue, Rosemary Monahan
Department of Computer Science
Maynooth University
Maynooth, Co. Kildare, Ireland
{diarmuid.odonoghue; rosemary.monahan}@mu.ie

Abstract
We describe recent work on the Aris system that creates and verifies new formal specifications for pre-existing source code. We describe Aris in terms of the abductive reasoning system that suggest possible specifications and then uses an existing deductive verifier to evaluate these creations. This paper focuses on the abduction system that creates new formal specifications by leveraging a small set of inspiring artefacts to augment a subset of candidate problems. This employs knowledge graphs to represent the raw data (i.e., source code), discovering latent similarities between graphs using a graph-matching process. Results are presented for the C# programming language with novel creations and its sister language called Code Contracts. We outline ampliative creativity, whereby newly created artefacts drive subsequent creative episodes beyond the initially perceived limitations. We also outline some recent work towards transferring specifications between the C# and Java programming languages.

Introduction
Formal specifications are central to adhering to safety standards and proving the correctness of mission-critical software, such as controlling nuclear reactors, vehicle and aircraft control, telecommunications infrastructure etc. Commercial sensitivity means that open-source specifications are not openly available, so specifications are typically created afresh. This paper addresses the challenge of creating formal specifications for existing source code, based on the likely intended functionality of that code.

The problem we address is to create formal specifications and proof requirements for given source code, akin to challenges in the VerifyThis (Dross, Furia, Huisman, Monahan, & Müller, 2021) series of program verification competitions. In these challenges, participants must create formal specifications, implement a solution, and formally prove that the implementation adheres to the specification. The formal specifications are written as a contract specifying the preconditions (requires clauses) that must hold, in order for the implementation to establish the postconditions (ensures clauses). The proof that the implementation adheres to the created specification is typically handled by an automated theorem prover which requires the users to provide axioms to assist the proof. These are written as assertions, invariants, variants alongside the source code. These specifications and axioms are often difficult for the user to create but writing them becomes easier with experience of the software domain. We are particularly interested in the creativity required in the use of both specifications and proof supports, ensuring the approaches adopted are transferable from one challenge to another (e.g., different sorting algorithms all require proof that the data is sorted and that the result is a permutation of the input data) and from one software verification tool to another (e.g., verification in Code Contracts for C# source code and Open JML for Java source code). A second challenge that software verification meets is poor uptake by industry (Huisman, Gurov, & Malikis, 2020). These challenges motivate us to investigate computational creativity to promote formal methods in the software-development process, automating some of the tasks involved in specifying existing source code. Our ARIS (Pitu, et al., 2013), (O’Donoghue, et al., 2014) system aims for professional (pro-c) creativity (Kaufman & Beghetto, 2009), comparable to professional formal-software developers.

We point out that formal specifications describe what an implementation achieves, while the implementation details how it is achieved. Specifications are often concise, describing the expected results of an operation. Formal theorem provers then verify the “what” against the “how”. The novel outputs of Aris are the problem implementation code and new formal specifications. Implicit in any new and useful artefact will be the newly verified theorem, uniting information derived from the implementation and specification.

We describe previously specified code as our inspiring set because our objective is that Aris can use all available proof strategies to specify a given implementation. Similar implementations can require different proof strategies, depending upon the specification language and the verification tool used. Aris aspired to support the full range of available proof strategies, tailoring the chosen strategy to the source code, specification and verification tool being used.
The remainder of this paper outlines of related work before describing the structure of the Aris system, describing how its abductive and deductive reasoning systems combine to create newly verified software. We describe the very small number of inspiring artifacts and a large collection of potentially solvable problems. We then describe our results, before showing how created artefacts can serve to drive subsequent creative episodes beyond the limits of the initial creative episode. Finally, we outline possible future work.

Background

Machine learning approaches to this problem are severely impaired by the chronic lack of specifications. We analysed several thousand projects from open-source repositories (SourceForge, GitHub etc.) containing over 2,000,000 methods and did not find any Code Contracts.

Recent work on systems such as GPT-3 and Text2App have shown some ability in creating executable code from a textual description. However, our problem does not have any such textual description leading us to rely solely on source code. A major problem with previous models of abduction (O'Donoghue & Keane, 2012) concerned the unreliable nature of their inferences.

There have been several recent advances related to code completion assistants. GitHub’s Copilot gives suggestions for lines of code or entire functions, taking as context any available docstrings, comments, function names, and the code itself. It’s recommended that Copilot’s outputs should be tested, reviewed, and vetted while in contrast, the output of Aris is evaluated by an automated theorem prover - as shall be discussed later in this paper. Interestingly, Copilot can also suggest test cases for a given implementation, however its ability to create formal specifications has not been reported. Copilot is based on Codex which is from the GPT family of language models, which is fine-tuned for code (Chen, et al., 2021). If sufficient specification were available train such a model (such as Aris can produce), these language models may become capable of producing specifications for given implementations. Despite these recent advances in code writing assistants, we are not aware of any system that can automatically add formal specifications to an existing C# implementation.

We argue that our challenge is more similar to the HR2 systems (Colton, 2012) that creates new mathematical theorems and is potentially capable of creating Prolog code. We think of problem code as containing facts and axioms that are known, with the objective of creating a new theorem (specification) likely to be useful in ensuring the correctness of that implementation. Source code and code contracts are translated into a form of first order logic and the Z3 SMT theorem prover takes the source code facts and the code contract “theorems” and attempts to verify one against the other. So, we can consider the code contracts created by our system as being akin to the theorems discovered by HR2. Aris also embodies significant differences to HR2 that created truly novel theorems, whereas Aris aims to create specification and proof supports that would be written by a competent professional formal software developer. Thus, historical H-creativity (Boden, 1992) is not required.

Figure 1 depicts the source code of a C# method for which we want to create a formal specification, highlighting a formal specification written as a precondition (i.e., requires) Code Contract1. The challenge for Aris is to create comparable formal specifications for similar implementations. The next section outlines the similarities that Aris can detect, as well as the mechanisms used to detect this similarity.

```
public char[] ReadNext(int count){
    Contract.Requires(0 <= count);
    char[] array = new char[count];
    for (int i = 0; i < count; i++) {
        array[i] = this.charBuffer[this.position++];
    }
    return array;
}
```

Figure 1: Problem code along and a Code Contract (highlighted) that we wish to create.

The ARIS System

Abductive reasoning excels at proposing hypotheses based on perceived similarity to some past scenario and has become associated with creative thinking. The downside of abduction lies in the reliability of its inferences and is sometimes accused of being grossly profligate in generating inferences even where no real similarity exists. In contrast, deductive reasoning excels at deriving definite conclusions from definite premises but is sometimes associated with narrow and constrained thinking. This section outlines how Aris creates new specifications using a combination of abductive and deductive reasoning to produce novel artefacts whose truth is mathematically assured.

We create specifications for source code written in C# with corresponding specifications in its sister language called Code Contracts (CodeCon). These act as a testbed for evaluating our bipartite creativity system. The core of this project is built on a general-purpose abduction system for discovering and extending similarities between general purpose knowledge graphs, being easily adapted for natural language and other data.

Extracting Code Graphs. The first process generates code graphs from the source code (Pitu, et al., 2013)\(^2\), using 18 categories of nodes (Variable, If, Block, Assign etc.) connected by 6 types of relations (Contains, Parameter, Returns that may additionally support co-creativity for related software artefacts including test cases.

\(^1\) Code Contracts were chosen for this project as it supports all .Net languages (C++, Python, Java etc.), it estimates the completeness of a partial proofs and works in an ecosystem
etc.) between nodes. Each method is described by its own code graph, whose nodes are extracted from the source code. Edges represent relationships between nodes, depicting static relationships extracted from the abstract syntax tree generated by the compiler.

Figure 2: Code Graph for the code in Figure 1 is best read from the top-level node “0: Block: Root”

Finding Homomorphic Graphs. Abduction from a solution to problem code requires perceiving some reasonable level of similarity between code graphs. We identify functionally similar methods by a process of homomorphic matching (Carletti, 2020) between code graphs, using the NetworkX library and initial experiments using its ISMAGS algorithms. Later experiments use a combination of topological similarity while also exploiting the identical labels between paired nodes (so a Block node should match a Block node in the other graph). Subsequent phases of Aris exploit mappings containing paired Variable nodes in the two graphs, as these represent potential destinations for adding new specifications.

Infer and Adapt. The next phases translates CodeCon from the source into the problem graph, after first checking the compatibility of the proposed inference. This requires identifying the relevant C# source code to find the corresponding variable type as this isn’t contained in the code graph. For example, if the CodeCon specifies `variable>=0`, this may be applied to both integer and real data types. This phase also uses the mapping to ensure the created specification interlocks with its new problem context. This phase also locates the correct point within the code to insert the CodeCon as this is essential for successful verification. This process uses the mapping and the source code of both methods to find the closest possible match to the original inspiring artefact.

Verify. Finally, the newly created code containing C# and CodeCon is added to its project for compilation, which automatically includes verification of any embedded CodeCon. Finally, all outputs are assessed to identify methods that were accepted by the deductive verifier.

Results and Discussion

We generated resulting using an inspiring set containing just 5 formally verified methods retrieved from the educational Rise4Fun website. Aris generated the code graphs and
iteratively attempted to abduce these specifications to a problem set containing around 1 million methods. While the ultimate objective is to create verified CodeCon, it is interesting to see how far along the Aris workflow each of the problems progressed as this gives us an indication of how like our process is to successfully extend to other source. Tables 1 and 2 quantify the number of candidate solutions as they progressed along the Aris workflow. “Potential targets” indicates the subset of methods selected for graph matching with that source. “Quality mapping” indicates the number of graph-matches above a hand-coded similarity threshold. The final two columns show the number of matches involving paired variable – a requirement for creating new specifications.

<table>
<thead>
<tr>
<th></th>
<th>Potential Targets</th>
<th>Quality mapping</th>
<th>1 mapped Variable</th>
<th>&gt;1 Mapped variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxDemo</td>
<td>145,824</td>
<td>98,323</td>
<td>62,189</td>
<td>12,468</td>
</tr>
<tr>
<td>ResizeDemo</td>
<td>11,142</td>
<td>9,845</td>
<td>5258</td>
<td>4,485</td>
</tr>
<tr>
<td>ResizeDemo Add</td>
<td>228,942</td>
<td>153,566</td>
<td>76,362</td>
<td>17,941</td>
</tr>
</tbody>
</table>

Table 1: Early Workflow Results

“Variable Type Match” in Table 2 indicates mapped variable have the same data type indicating a potential target for that specification. “Adapted” indicates that a new specification was created after adapting the inferred specification to better fit its new problem context.

<table>
<thead>
<tr>
<th>Variable Type Match</th>
<th>Adapted Potentially Verifiable</th>
<th>Successful Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>489</td>
<td>2,111</td>
<td>489</td>
</tr>
<tr>
<td>786</td>
<td>6987</td>
<td>786</td>
</tr>
<tr>
<td>2154</td>
<td>72,688</td>
<td>2,154</td>
</tr>
<tr>
<td></td>
<td>3,429</td>
<td>11</td>
</tr>
</tbody>
</table>

Table 2: Late Workflow Results

“Potentially Verifiable” indicates specifications that we expect could successfully verify if we were able to reconstruct the entire project. Unfortunately, we were unable to verify many specifications because of incomplete repositories, code being incompatibility with the compiler version required by CodeCon and unavailable libraries.

The final two columns indicate successfully verified CodeCon. Aris used three specifications to create over 3,400 potentially verifiable new specifications, 11 of which were successfully verified. “Contract Density” is the deductive verifiers assessment of the completeness of that verification being slightly above 50%, indicating additional CodeCon are required for complete (100%) verification and an assurance that this source code can never yield an unexpected result.

Figure 2 depicts the typical distribution in similarity between a chosen graph and candidate graphs from the corpus. The vertical axis indicates the percentage of edges from the specified code graph that have been matched with a potential problem graph. The horizontal axis lists the similarity scores for a selection of similarly sized graphs, which here have been sorted according to that similarity score. The minimum level of similarity required to support creative abduction is an open question.

**Static graphs.** Our results are even more surprising as the static code graphs do not contain information on the relative ordering of statements within each Block – that is, there is no information indicating which statements occur first, second etc.

**Self-Driving Creativity**

Next, we see how a newly created artefact listed in Table 2 served to increase the creative ability of Aris. Because deductive creators can pinpoint the source of their creative leaps, we now show how a created artefact was used to drive subsequent creative inference. A graph was generated for the newly verified code and compared to the corpus. Figure 3 shows a problem method and highlighted (in yellow and boldface) is a newly created specification. Again, this method was successfully verified against the specification.

```csharp
public static IEngineConfigurationTypeBuilder<TPoco> Value<TPoco, TMember> memberConfig, TMember value) {
    return memberConfig.Use<ValueSource<TMember>> { new object[] {
        Value });
} }
```

Figure 3: The CodeCon in this code was created by Aris, using one of its own created artefacts.

Due to Aris’ use of a graph matching process, the created artefact may match artefacts that did not match the initial artefact. Thus, a created artefact may increase the creative abilities of this abduction-based model.
al., 2014) this is referred to as self-sustaining creativity where a created artefact serves to extend the creative potential of the system beyond the originally perceived limits. Figure 3 shows a “generic method” that was matched with a previously created specification.

The extreme lack of available specifications negatively impacts the diversity of the validated CodeCon. This lack of specifications can be addressed by a) manually creating more specification or, b) by looking to other languages as a possible source of specifications.

Between C# and Java
To find more formal specifications we next look to programs written in Java where the corresponding specifications are in its OpenJML sister-language. A corpus of around 10,000 code graphs was generated from open-source Java repositories. Figure 4 shows a Java method identified as similar to that of Figure 3 above, with this Java code clearly having much additional information. It is hoped we can explore bi-directional comparisons between languages, allowing transfer of specifications between them. We expect this approach to offer some additional CodeCon, however differences in coding language, specification language and underlying theorem provers may impinge upon the verification of some specifications.

Future Work
We expect that our current results may be improved by improving graph matching process and placing a greater focus on mappings between variables, as this may lead to generating more usable specifications. The Aris project may also allow us explore interactions between newly created artifacts and the verifier system (the Z3 SMT solver), because it can sometimes discover additional CodeCon that may help a human (or artificial) user to increase the CodeCon density of the solution. CodeCon can also generate a suite of test cases(data) using the PEX tool, further extending the range of items created from Aris specifications. However, there is currently no transferable learning between its creative episodes.

Conclusion
We describe a combination of abductive and deductive reasoning that suggests and then formally verifies new formal specifications for some given problem implementations. Aris detects similarity between static knowledge graphs derived from source code and creates formal specifications that were successfully verified. Furthermore, specifications created by Aris served to drive subsequent computationally creative episodes. The fundamental limits to self-sustaining creativity in this context remain to be explored.

References
In the Name of Creativity: En Route to Inspiring Machines

Chun-yien Chang*
College of Humanities and Social Sciences
National Yang Ming Chiao Tung University
HsinChu City, TAIWAN
ccy@cctcc.art

Ying-ping Chen*
Department of Computer Science
National Yang Ming Chiao Tung University
HsinChu City, TAIWAN
ypchen@cs.nctu.edu.tw

Abstract
In this short paper, we reflect on the long quest for intelligence and creativity of computing machinery as well as the suitability for prevailing machine learning techniques to be used in creative tasks. We believe that modularization and multi-layered structures are among essential ingredients constituting creative minds and may greatly benefit machines on handling creative tasks. For proof of concept, we select musical composition, particularly, Species Counterpoint, as the task, adopt a recently proposed computational framework designed for investigating creativity and the creative process, and present an implementation capable of producing the outcomes that exhibit the desired effect.

Introduction and Reflections
Computing machinery has been fascinating to human beings for quite a long time. Accompanied with the introduction to the idea and design of a programmable, general-purpose, mechanical computer, well known as Babbage’s Analytical Engine (Menabrea 1843), almost two centuries ago, expectations and speculations on the potentials, especially in aspects of intelligence and creativity, of such a machine had been boldly made by Lovelace (1843), “…, the engine might compose elaborate and scientific pieces of music of any degree of complexity or extent.” A century later, Turing (1950) asked the question, “Can machines think?” to address the intelligence aspect of machines and to argue that machines may eventually exhibit intelligent behavior as playing well in the imitation game. Now, we wish to ask the question, “Can machines create?” Under the current circumstances, we are unable to directly, appropriately answer this question. Instead, in this article, within the scope of musical composition, we would like to make a discussion on the apparent lack of certain essential components, capabilities, and properties that enable or permit machines to create in the present prevailing techniques. Moreover, we provide our preliminary implementation as a viable technical construction with its generated results indicating that the existence of some these essential ingredients brings machines one step further closer to being able to create.

The most prominent, prevailing computational techniques in the related fields of artificial intelligence are undoubtedly the methods in the family of deep learning and artificial neural networks, and in the area of music generation, there have already been enormous studies and results (Briot, Hadjeres, and Pachet 2020). We do not intend to diminish the importance or undermine the practical value of those works, but if methods of this category are adopted, from the viewpoint of creators, or more specifically, music composers, in terms of the present form of the methodology, there are certain limitations in the aspects of formality, capability, and efficacy (Pearl and Mackenzie 2018). The technical framework of deep learning requires a huge amount of data, i.e., existing music pieces in this case, to train models which no matter will be used as classifiers or generators. Insufficient data will be unable to render useful or meaningful outcomes, let alone models that can create in the common sense. In history, there are only a few productive music composers creating certain amount of music pieces. Simply according to this situation, in contrast to the usage and requirements of the deep learning methodology, it can be seen that creativity and the action of creating may not properly fit with how deep learning operates and functions.

As to the characteristics of deep learning, please allow us to make an arguable analogy. We know that any boolean function, as long as the truth table is given, in theory, we can directly construct its combinational circuits in a systematic way by analyzing and identifying the essential prime implicants. It is in the fundamentals of logic design. However, in practice, except for certain, usually extremely simple, circuits, most circuits are not designed in this way. Instead, their design process usually incorporates modularization, multi-layered structures, domain knowledge, and even personal experience of designers. Although the techniques of deep learning keeps evolving and advancing, employing popular deep learning techniques on music generation tasks is intrinsically similar to making attempts to piece together superficial elements, like prime implicants in the circuit case, to produce target outcome in the basic, primitive way. For music, such an approach in fact not only conceptually ignores the separate, usually totally different ideas and emotions that the music composer would like to convey and express via individual music pieces but also decontextualizes the music pieces by not considering the essence constitutes the creation such as the historical background, the cultural heritage, and even possibly the factors of instruments, including timbre, registers, and the difficulty to perform.
Moreover, pre-existing knowledge, pre-determined settings, and personal preference or experience are extremely difficult to inject into the use of deep learning methods if at all possible. While the models obtained from deep learning can be presented in detail in the form of many parameters, usually millions, and easily duplicated for replicating the results, the operation as a whole fundamentally forms a black box. Thus, a successful, practically applicable artificial neural network model can provide little information for gaining insights or triggering inspiration. In recent years, while the research directions such as interpretable machine learning and explainable artificial intelligence have emerged (Linardatos, Papastefanopoulos, and Kotsiantis 2021), the advancement is currently quite limited.

Therefore, in this article, we wish to respond to the call made by Turing for making machines intelligent, or in our case, capable of creating. We would like to take a small step towards making machines able or seemingly able to create in the common sense. In order to integrate the computational framework with the concept of modularization and the multi-layered structures of the creating process, we adopt our recently proposed meta-framework, ants on multiple graphs, AntsOMG (Chang and Chen 2020) and one of its showcase, the composition of organum motets (Chang and Chen 2021), hence, in the hope that the essential ingredients in the creating process, especially in music composition, can be observed. Based on the design and properties of AntsOMG, we expect the presented implementation to possess certain characteristics, such as accessibility, scalability, and explainability. Moreover, at the level of technical details, since the implementation is multi-layered and modularized, the “components” of the produced model for composing music can even be separated and swapped with ease. Hence, transfer learning, utilizing pre-existing knowledge, incorporating pre-determined settings, integrating human experience, and the like can be achieved. By conducting research along this line, hopefully injecting creativity into machines may someday be accomplished.

### Related Work

While the goal of this study is to investigate creation behavior and mechanisms, in the hope that a small step towards enabling machines to autonomously create may be accomplished, automated music composition is closely related to this article since a particular music genre, organum motets, is adopted as the study subject. Hence, selected studies available in the literature in the realm of computational intelligence related to music generation and automated composition are included in this section for reference. More comprehensive surveys and reviews can be found by (Loughran and O’Neill 2020; Carnovalini and Rödén 2020; Liu and Ting 2017; Herremans, Chuan, and Chew 2017; Lopez-Rincon, Starostenko, and Martín 2018; Briot, Hadjeres, and Pachet 2020; Gifford et al. 2018; Fernández and Vico 2013).

Evolutionary algorithms are population-based, stochastic optimization methodologies relatively easy to be used to handle a variety of tasks of very different nature. Because of their flexibility and versatility, they have been utilized for generating music decades ago. Genetic algorithms, one of the major evolutionary algorithms, have been used in the task of computer-assisted music composition (Horner and Goldberg 1991; Jacob 1995; Marques et al. 2000). More complicated music constructs are also considered by researchers, including chord progression (Kikuchi and Otsana 2014), measures and phrases (Ting and Wu 2017), harmonization (Donnelly and Sheppard 2011), and music theory (Liu and Ting 2012).

For other branches of evolutionary algorithms, genetic programming has been applied to evolve music generation (Phon-Amnuaisuk, Law, and Kuan 2009) and to compose the 16th-century counterpoint (Polito, Daida, and Bersano-Begy 1997). Ant colony optimization has also been adopted to generate music (Guéret, Monmarché, and Slimane 2004) and to create Baroque harmonies (Geis and Middendorf 2008).

### Proof of Concept and Outcomes

Aiming at enabling machines to create, as aforementioned, we select classical music as scope, in particular, Species Counterpoint. We will firstly give some background regarding Species Counterpoint as it is closely related to what we would like machines to have, what role such a mechanism plays in the creative process, and what effect can be observed when machines have such a capability.

During the period of conventional music theory, approximately from Johann Joseph Fux’s famous textbook *Gradus ad Parnassum* (Fux 1725) to its decline after Franz Schubert (Mann 1994), Species Counterpoint has been an essential way to pursue the ability of composition. Even the great composer Ludwig van Beethoven also learned Species Counterpoint from Joseph Haydn and left many manuscripts of his studies (Mann 1970). It is also a well-organized and constructive way to approach the composition of polyphonic music. Fux’s title *Gradus ad Parnassum* (Steps to Mount Parnassus) revealed the step-by-step essence to construct the required knowledge for the sixteenth-century counterpoint. Species Counterpoint is not only a step-by-step guide, it also exhibits the building blocks essential to a classic type of creative behavior:

- **Monophony**: The structural rules for both cantus firmus and counterpoint melodies;
- **First Species**: The backbone of all contrapuntal interactions consists of note-against-note consonant intervals;
- **Second Species**: Introducing the usage of dissonance (passing tones) and the concept of strong-weak beats;
- **Third Species**: Introducing more types of dissonances including neighboring tones and cambiatas, and more differentiated contrapuntal layers and rhythmic structure;
- **Fourth Species**: Introducing the most important type of dissonance at that time (suspensions), and the translocation of strong beat by shifting the notes;
- **Fifth Species**: Integration of all previous Species to achieve a fully organic unfolding of melodic beauty in the context of counterpoint; . . . and more.

---

*Proceedings of the 12th International Conference on Computational Creativity (ICCC ’21)*

ISBN: 978-989-54160-3-5

401
Figure 1: An example of black-box AI. Google (2019) launched their first AI-powered Doodle Celebrating Johann Sebastian Bach on March 21, 2019. We use Bach’s own chorale melody fragment from “Ach wie flüchtig, ach wie nichtig” (Bach 2008) to get the results of harmonization twice: The first result contains a bass line that is very awkward for human voice and also a strange chord in the beginning of the second measure; the second result comprises the uncompleted inner voices which are not even correctly notated. It demonstrates the problems of black-box AI: the machine still lacks any basic ideas of Bach Chorales even after receiving the so-called training.

This definite building process does not limit the development of composers, like Haydn or Beethoven, but rather serves as the foundation of their exploration of distinct personal styles. This makes us reflect on whether it is sufficient to solely regard the black-box machine creativity as the mainstream methodology while exploring computational creativity, an even broader realm of artificial intelligence. What are the inner foundations the composers have built while learning the tedious counterpoint rules, so that they not only gain the ability to write the counterpoint exercises like a machine but also develop their own styles upon it?

In order to explore this topic, we began to construct the building blocks of Species Counterpoint and to make attempts to “inject” this longlast heritage of music creativity into machines in a white-box way—different from blackbox AI—to explore the different possibilities of machine creativity. (See Figure 1)

To demonstrate the different possibilities of machine composition, we formulated a musical form with elements from First Species Counterpoint, medieval organums, and Renaissance motets and implemented a computational framework to enable machines to compose a full length musical piece with its own compositional inclination.

First, we implemented the algorithm to generate the plainchant based on the rules of the following melodic intervals (Jeppesen 1992):

- Ascending and descending: major and minor second, major and minor third, perfect fourth, perfect fifth, and perfect octave.

- Ascending only: minor sixth.

For Renaissance vocal counterpoint, a melody is not only a series of numbers but also a medium suitable for human voice singing and conveying religious feelings. Therefore, we constructed the algorithm operating with multiple graphs: a gamut graph derived from the pitch set (Figure 2) and the aforementioned melodic interval rules as well as a meta graph representing the thinking process of composing melodies to regulate the output from the gamut graph.

Next, based on this algorithm, we implemented the most important rules of harmonic intervals from First Species Counterpoint: (Jeppesen 1992)

- Only consonant combinations may be used (the fourth is considered a dissonance).

- One must begin and end with a perfect consonance (octave, fifth, and so on). However, if the counterpoint lies in the lower part, only the octave or unison may be used at both beginning and the ending.

We expanded the gamut to allocate the upper and lower counterpoint parts (Figure 2) and transformed it with the First Species harmonic intervals rule set and the assigned cantus firmus into a new graph for counterpoint composition. Figure 3 is an example of a musical excerpt from one of the generated compositions.

In this article, we intend to demonstrate the “Neo-classical” fun that we discovered over the process of the injection of creative knowledge. In the early twentieth century when many creative methods flourished, Neo-classicism is among the most remarkable ones, which is somewhat sarcastic in the interwar atmosphere. Russian composer Sergei Prokofiev’s comment “Bach on the wrong notes” (Tierney 1977) on his fellow countryman Igor Stravinsky (See Figure 4), vividly and arguably indicated the impression of Neo-classical works on people. When Prokofiev composed his “Classical” Symphony, he used an interesting metaphor: “It seemed to me that if Haydn lived to our day he would have retained his own style while accepting something of the new at the same time. That was the kind of symphony I wanted to write: a symphony in the classical style.” (Prokofiev and Shlifstein 2000)
Figure 4: Music excerpt from Movement I of Stravinsky’s Concerto for Piano and Wind Instruments (Piano solo part, from measure 49). Here it exhibits a taste of Bach’s three-voice works, except that it is “on the wrong notes”.

Figure 5: An example of a phrase of an organum motet generated from the replaced gamut graph, based on the cantus firmus in Figure 3.

Taking one’s intelligent and emotional experiences as a foundation, while constantly pursuing the innovation of creative elements, is the lifelong effort of the so-called “serious” music composers. However, just take the example of the combination of pitch sets, when a human composer has gone through the process from being a student, all the way to the point when a revolution becomes apparent, this person has accumulated a vast amount of experience on composition. In other words, a strong and inevitable tendency has been formed in the years over one’s development, it is never easy to switch like an instantly replaced component. This may be one of the reasons why Neo-classicism has seemed attractive and challenging to many twentieth-century composers. Most of them have more or less ventured the route of Neo-classicism, regardless of what their own mature styles may be.

For machines, the switching of algorithmic components is undoubtedly one of their strengths, as long as they are constructed or expressed in the form of a white box. Take the first species counterpoint as an example, under the identical harmonic and melodic interval rules, swapping the gamut graph and the corresponding score functions renders interesting results, as shown in Figure 5, of a very different style.

Over the presented process, we adopted a computational framework, called AntsOMG, specifically developed for investigating creativity and the creative process. AntsOMG inherits certain characteristics of ant colony optimization, ACO (Dorigo and Gambardella 1997), letting machines develop a set of style models through a large quantity of practices, and we slightly loosen some of the rules to amplify this tendency to allow ample space for development. Our implementation is not limited only to produce counterpoint fragments but to create a complete musical piece called organum motet, which comprises several segments of plainchants and counterpoint based on them. Thus the aforementioned swapping is also applicable to the generation of the complete organum motets—with a stylistic twist. More examples of both types of organum motets are provided in the supplemental material.

While retaining most of the building blocks and the style development mechanism, we have changed the creative phenomenon produced by machines merely by swapping a single graph, which may be intriguing in the context of transfer learning because of the nature of modularization and its multi-layered design. As the present form of AntsOMG, the weights/parameters on the developed model, unlike those on artificial neural networks, are quite interpretable, and consequently the behavior of the whole algorithmic construct may be explainable. It is a simple display of the benefits of injecting white-box knowledge into machines. In the meanwhile, it opens up many possibilities of machine creativity through the unlimited combinations or swapping of the building blocks, such as the pre-existing or newly generated rule sets, creative materials, and even the outcomes from the interactions between them. The diversity of computational methods and their combinations, along with the current achievement of machine learning, will largely enrich the field of computational creativity, and even human creators may further benefit from being inspired by the plentiful spectrum created from the landscape of machine strengthened creativity.

Supplemental material of this paper, including scores, music, and source code information, is available and can be accessed online via https://e.cctcc.art/iccc21 or https://github.com/nclab/iccc-21.

Conclusive Notes
In this article, we presented our ideas and thoughts on what machines must have in order to possess the capability of creating. Human creators learn the domain knowledge, gain their personal experience, and accordingly cultivate their own styles during the time of their existence. Machines are intrinsically different from humans and do not need to act like humans. However, before machines are able to independently, without any human help or dictation, exhibit a sufficient level of intelligence, investigating how creative minds operate and gaining insights into the creative process are necessary to make progress towards inspiring machines.

Acknowledgments
The authors would like to thank Taiwanese renowned cellist Dr. Sheng-wen Liu for her professional advices and Mr. Chih-fang Chiang for his assistance with the experiment data processing and music rendering. The work was supported in part by the Ministry of Science and Technology of Taiwan under Grant MOST 110-2221-E-A49-076.

References


CREA.blender: a GAN based casual creator for creativity assessment

Miroslav Gajdacz1, Janet Rafner1, Steven Langsford1, Arthur Hjorth1, Carsten Bergenholtz1, Michael Mose Biskjaer2, Lior Noy3, Sebastian Risi4, and Jacob Sherson†1

1Center for Hybrid Intelligence†, Department of Management, BSS, Aarhus University, Denmark
2Center for Digital Creativity, dpt. Digital Design and Information Studies, Aarhus University, Denmark
3Business Administration, Ono Academic College, Kiryat Ono, Israel
4IT University of Copenhagen, Copenhagen, Denmark

Abstract
In this technical demonstration paper we document the use of a Generative Adversarial Network (GAN) based casual creator game for systematic assessment of human creativity. We discuss some of the challenges in designing GAN-based casual creators, specifically focusing on how to identify and select appropriate parts of the latent vector space - images, in our case.

Introduction
Since their invention, Generative Adversarial Networks (GANs) (Goodfellow et al. 2014) have deservedly attracted attention in the field of computational creativity (Berns and Colton 2020). GANs excel at producing realistic artifacts (Karras, Laine, and Aila 2019) and have the potential to replace humans at the most laborious parts of the manual creative process, to the extent of facilitating visual creation for people without training in the manual arts.

GANs are well-known for producing artificial images that can be nearly indistinguishable from real images (Borji 2019). More recently, artists have begun using GANs for creative purposes to create images and music (Berns and Colton 2020). However powerful these tools may be, they come with substantial requirements for the technical expertise of their users to be able to implement and execute the underlying Machine Learning models and devise the processes, which create the desired artifacts.

In a recent trend called ‘casual creators’, digital tools are designed to “empower autotelic and enjoyable amateur creativity” (Petrovskaya, Deterding, and Colton 2020). Casual creators afford the creation of highly elaborate artifacts with little input from users. At the core of these products often lie algorithmic generators, of which GANs are just one example. These generators map from a simple low-dimensional input domain, such as sliders, dragging gestures, numerical or multiple choice parameters, which are relatively easy to understand for the users, to a complex high-dimensional output domain, such as images or audio. In a well-designed casual creator, the mappings are intuitive enough to allow for meaningful directed search, creativity, and serendipity to emerge from people’s interactions with these systems.

Thus far, casual creators have mostly been used for entertainment (Petrovskaya, Deterding, and Colton 2020). In this paper we propose a novel use of casual creators: to systematically assess human creativity. Standard psychometric tests for assessing creativity are based on simple tasks such as making as many unique drawings as possible out of circles (Torrance 1966) or coming up with alternative uses for an object (Guliford 1968). This design is intentional so that the task is widely accessible. However, in most circumstances it does not allow for complex, interesting expressions of ideas. Utilizing a casual creator has the potential to change this.

Here, we present crea.blender, a GAN-based image generation game, explicitly designed to assess creativity of the general public (Rafner et al. 2020). This game is part of a broader suite of games and tasks to measure creativity, called CREA (Rafner 2021). crea.blender utilizes a BigGAN model (Brock, Donahue, and Simonyan 2019) in three distinct sub-tasks where a set of base images, either predefined or selected by the user, can be "blended" together into one output image.

The design of crea.blender draws heavily on existing casual creation systems, particularly Artbreeder (originally Ganbreeder) (Simon 2021). The crea.blender system is distinct in two main respects. Firstly, the crea.blender interface is somewhat simpler and does not expose individual ‘genes’ to player manipulation as Artbreeder does. More importantly, crea.blender is divided into three play modes, the challenge mode, divergent mode, and open play mode, in order to expose different elements of the creative process to systematic measurement.

Details of the three play modes and their theoretical foundations are given below, but in brief the challenge mode asks players to reproduce a target image, the divergent mode asks them to create as many different images of a particular theme, and the open play is unconstrained. A core goal of crea.blender is to explore the extent of the relationships between the different modes. Some elements of the task, like the perceptual similarity between images, are unlikely to change across modes. Others, like the search strategies employed, could potentially change drastically in response to the incrementally looser constraints. By applying different task demands to the same underlying space, crea.blender offers the opportunity to look for distinctive features of these strategies, and the extent to which these features recur across
the different tasks types. We consider the extent to which these different tasks can be understood as reflecting overlapping cognitive abilities to be an important open question.

The purpose of this demo article is to outline a novel use of casual creators as a tool for systematically assessing creativity, and to discuss some of the fundamental design challenges in creating interesting creativity tasks with a GAN. We do this by providing a technical description of the GAN model and how it is utilized for cross-category image blending. We then outline and discuss challenges relating to finding suitable images which provide sufficient expressiveness in the creative process for the non-specialist users. Finally, we discuss more general implications for designing casual creators with GANs and generative ML.

The Game
CREA is a game written in Unity and runs via WebGL in a browser, making it accessible on a wide range of desktop and mobile devices. The execution of the underlying GAN model involves a large amount of parallel calculations, which can be significantly sped up (by factor 10-100) when performed on a graphics card (GPU) compared to CPU. To broaden participation and make crea.blender accessible for people without GPUs, the image generation is performed remotely on a server with a GPU. Our GAN server is running on an Azure Virtual Machine and can process around ten image-generation requests per second.

Challenges in designing with a GAN
The GAN model (Brock, Donahue, and Simonyan 2019) has been trained on ImageNet (Deng et al. 2009) and uses as input a 1,000-element class vector and 128-element noise vector, which together form a so-called latent vector. Each class represents a real-life object, i.e., a beetle, a dog, etc., and each specific latent vector, when propagated through the GAN, deterministically produces an image.

The original purpose of the GAN is to generate realistic examples of a single chosen class, which is done by inserting randomly generated instances of the noise vector, while keeping only one component of the class vector non-zero (the selected class).

Image mixing procedure
In crea.blender we allow for multiple class vector components to be non-zero, therefore the output images rarely resemble members of any particular class. Rather, each of the images that people use are combinations of many different real-life objects. The base images that are entering the mixing were also produced by the GAN and are each specified by their latent vectors.

The players control the mixing procedure by sliders located below each base image. Once they have chosen some desired slider settings, they push the image generation button, which sends an image generation request to the server. The latent vectors of the base images are linearly superimposed (added together) with weights proportional to the slider values. Since we do not perform normalisation of the combined latent vector, adding the images with small weights can lead to unusual outputs not resembling the base images.

Design challenges
The fundamental challenge in creating a causal creator with a GAN is selecting the images that users can blend together. The images should be interesting and have aesthetic qualities in their own right. But more importantly for a casual creator like ours, the underlying latent vector of these images should ‘blend well’ with the other images, meaning that if you take two images and blend them, the resulting image should meaningfully look like a combination of the two. Further, the different creative modes have slightly different requirements for their images, discussed below.

Modes of creativity assessment
crea.blender has three creative modes, each one utilizing the GAN in a different manner. We aim to measure and study player performance across tasks that vary in the specificity of their goals. This is to assess the players ability in:

- Expressing themselves, that is reaching a specific target: Challenge mode
- Producing many alternative solutions to a specified theme: Divergent mode
- Producing novelty and value in general: Open-play mode

Challenge mode
The design of the challenge mode is built to assess the creative process convergent thinking, which is defined as the ability to find the single best solution to a defined question (Guilford 1956). Each trial in the Challenge mode has two stages. In the first, participants are presented with three sets of three images (see Figure 1.a), on the left) and have thirty seconds to indicate which of the three sets can produce a target image (see Figure 1.a), on the right). After they have selected the correct set of images (possibly on the second or third attempt) participants progress to the blending stage, and attempt to reproduce the target image by setting contribution sliders appropriately on the three base images (see Figure 1.b). The participant-generated image is updated whenever the generate button in the center of the screen is clicked. The trial ends when a generated image is sufficiently close to the target or when two minutes have elapsed in the blending stage. Feedback is given with text prompts at the end of each stage.

Divergent mode
The divergent mode is built to assess ‘divergent thinking’ a creative process which can be defined as the ability to come up with many different solutions to a prompt (Guilford 1956). Divergent thinking is often further broken down into the components of ideational fluency (the number of outputs: ideas, products, solutions), flexibility (how different the proposed outputs are from each other), originality

1Acceptance thresholds were manually specified for each image.
Figure 1: Challenge mode: a task with a well defined goal. a) Choose one image set which can produce the given target image on the right. b) Chose the image mixing weights for the base images (sliders below) to produce a blended image (on the top left) such that it is close to the target image (on the top right).

Figure 2: Screenshot of the game in the Divergent mode. The Open-play mode looks identical, but allows for switching the individual base images by clicking on them.

Figure 3: Examples of images that can be generated in the Open-play mode.

Open-play mode

The open-play mode asks participants to “create as many creative images as you can” in four minutes. An option to end the task early is available after five images have been submitted. Visual presentation of this task is the same as in the Divergent mode, but the set of base images is not fixed. Clicking on a base image replaces it with another base image drawn randomly from a pool of 33 items. This new mechanic is introduced to participants with a short tutorial at the beginning of the task. The open play mode is assessed through based on the commonly accepted definition of creativity: novelty and value (Runco and Jaeger 2012). Currently the images are assessed through crowdsourcing of other participants in an evaluation phase, but we are working on supplementing this with algorithmic techniques such as clustering.

Pre-selection of the base images

The image mixing process on a pre-trained GAN is relatively simple. The difficult part is to provide users with aesthetically pleasing base image sets, which do not produce offensive or distasteful outputs when blended, e.g. weirdly disfigured creatures. The output of the GAN can sometimes be quite unpredictable, especially when the slider weights are set low. As a case in point, we have seen that a mixture of an image resembling a mango with an image resembling a shower head can give a rise to a dog, a cat or a squished human head. Weeding out such base images can be a tedious manual process, which we have partially alleviated by creating systematic line scans of the latent vector space for the different base image sets and glanced at the results by watching a fast-paced movie compiled from the output images.

Another issue in the selection of the base images is the support of creative intent. Users should be able to some extent predict the output of the GAN. When mixing images containing multiple objects or features, it is hard to judge

which features will make it into the resulting mix. Gener-
ally, we observe that difficulty of predicting the GAN output
depends on the following factors:

- The number of base images provided
- The number of base images used in the target blend
- Presence of characteristic features in the base images that
can be identified in the target image
- Distinctness of features across the alternative base image
sets
- The relative volume of the parameter space that produces
something similar to the target (provided the correct base
images are blended)

**Challenge mode image sets**

In order to obtain suitable images for the Challenge mode,
we have developed and applied a Monte Carlo-style algo-
rithm, which generates random examples of basis image
candidates and then evaluates them by simulating their use
in blending. In order to reduce the count and location of
distinct features in the base images, we require that a major
fraction of the output images has:

- Uniform background: limit on the maximum color differ-
ence between selected pixels along the image border
- Smooth background: upper limit on the average Sobel
gradient magnitude (density of sharp edges) in the border
region
- Good foreground contrast: lower bound on the color dif-
ference between the background and selection of centrally
located pixels

Once we have obtained a sufficiently large pool of base
images, we have clustered them by the color of their back-
ground using the k-means algorithm with three clusters. We
then draw randomly one image per cluster for each of the
three alternative image set, which ideally produces balanced
(similar) challenge item sets. In practice, manual selection
was involved afterwards, to swap base images with inappro-
priate or very obvious features. We have also performed sys-
tematic slider space scans to ensure no majorly disturbing
images arise when the users explore the space.

**Discussion**

GAN systems are particularly well-suited to producing rich
and complex outputs from a relatively simple interaction.
Although there are significant design choices involved in
setting up such a system for casual creators, as described
above, users of the system can produce a highly diverse ar-
ray of possible outputs even with just three to five base im-
ages. Most importantly for assessment purposes, they do so
with a constrained set of tools (in this case slider manipula-
tion and base image selections) which, unlike paintbrushes
or chisels, can be wielded in much the same way by almost
anyone.

A primary advantage of the ease-of-use of such a GAN
system is how the basic interaction can be used in a range of
task designs targeting specific cognitive components of cre-
ativity. Here, we have begun to exploit this by designing a
series of tasks that vary in the specificity of the goal, allow-
ing for a contrast between abilities supporting open-ended
creativity and more goal-oriented creative tasks. These
are relevant to studying divergent and convergent thinking
(Guilford 1956), which may be supported by a common
set of cognitive abilities, but are also considered to disso-
icate under some conditions (Chermahini and Hommel 2010;
Chermahini and Hommel 2012). A deep dive into how di-
vergent and convergent thinking are operationalized in the
CREA suite can be found here (Rafner 2021)

Another advantage of a casual creators is their accessibil-
ity to a broad audience. Widening participation is impor-
tant for testing subtle effects that require large participation
numbers to be detected reliably. Additionally, and perhaps
more importantly, it is a crucial property for creativity as-
essment tools as it must be broadly usable and capable of
supporting creativity without relying on craft-specific com-
petencies. crea.blender meets these requirements, since par-
ticipants only need to choose some base images and manip-
ulate their associated sliders in order to create a vast range of
distinct artefacts and explore the complex high dimensional
output space.

We hope crea.blender will help pave the path for the use
of casual creators in studying creativity at scale and making
GAN-based generators accessible to the general public.

**Acknowledgments**

We would like to thank the Carlsberg Foundation, Novo
Nordisk Foundation, and the Synakos foundation for their
generous support of this work. Additionally, would like to
thank the CREA consortium for meaningful conversations
regarding this work and the development team at SAH for
building the CREA suite.

**References**

Bridging Generative Deep Learning and Computational Cre-

[Bojji 2019] Bojji, A. 2019. Pros and cons of GAN evalua-
tion measures. Computer Vision and Image Understanding

[Brock, Donahue, and Simonyan 2019] Brock, A.; Donahue,
J.; and Simonyan, K. 2019. Large scale GAN training
for high fidelity natural image synthesis. arXiv preprint:
1809.11096.

[Chermahini and Hommel 2010] Chermahini, S. A., and
Hommel, B. 2010. The (b)link between creativity and
dopamine: spontaneous eye blink rates predict and
dissociate divergent and convergent thinking. Cognition

[Chermahini and Hommel 2012] Chermahini, S. A., and
Hommel, B. 2012. Creative mood swings: divergent and
convergent thinking affect mood in opposite ways. Psychol-
ogical research 76(5):634–640.
Dynamic Creation of Points of View

Alessandro Valitutti
Phedes Lab
http://phedes.com
alessandro.valitutti@phedes.com

Abstract
We present a proof-of-concept prototype aimed at providing a subjectivity touch to the production of creative artifacts. The underlying idea is to have a system in which the development of the artifact occurs concurrently with the dynamic formation of a personal identity, thus enriching the overall perception of creativity.

Introduction
Computational generation of creative artifacts, such as artificial painters, music composers, and poetry generators are mainly focused on the artifact itself and the properties it can exhibit as part of its creative value. However, there are several tasks where the perception of a coherent persona can improve the creative quality of the artifact or the interaction. For instance, in computational storytelling, it could be the simulation of the point of view of the author or that of one of the characters (Diasamidze, 2014; Al-Alami, 2019) to provide believability to the narrative (Riedl and Young, 2010; Kybartas and Bidarra, 2016). In the context of human-computer interaction, persona’s believability improves the interaction with a chatbot or a personal assistant (Demeure, Niewiadomski, and Pelachaud, 2011).

In this paper, we present a proof-of-concept system aimed at providing a subjectivity touch to the production of creative artifacts. Specifically, we propose an architecture in which the generation of the object (i.e., artifact) and the development of the subject (i.e., the dynamic personal identity) are designed as concurrent processes that both contribute to creativity of the system.

Background
This section reports the main conceptual distinctions used as a source of inspiration for the design and the development of the proposed system.

Self-Aware Computing
Self-aware computing is a recent area of computer science concerning autonomic computing systems capable of capturing knowledge about themselves, maintaining it, and using it to perform self-adaptive behaviors at runtime (Lewis et al., 2015; Torresen, Plessl, and Yao, 2015; Amir, Anderson, and Chaudhri, 2004). Autonomic computing is about finding ways of managing computing systems reducing their complexity by making individual system components self-managing, thus reducing the need of human maintenance. In this research we reinterpret one of the characteristics defining autonomic systems, such as self-awareness, in the context of computational creativity (Hariri et al., 2006).

We can see a self-aware computing system as a program provided with a runtime extension of a target system (in a similar way a debugger program works enriching the execution of a procedure) and characterized by (self-)access (i.e., the introspective access to the execution process), (self-)modeling (i.e., the development of a model of the target), and (self-)expression (which refers to either the runtime modification of the target, informed by the model, or reporting, as in the case of self-diagnosis systems for autonomous vehicles (Jeong et al., 2018)).

Computational Reflexivity
 Reflexivity is a specific type of computational self-awareness (Valitutti and Trautteur, 2017). During the reflexion process, the three components of self-awareness introduced above (access, modeling, expression) are not only meant to be occurring in runtime but also in a strict concurrency. As a consequence, the target process and its reflexive augmentation are functionally distinct, yet continuously interdependent.

It is worth to point out that, while a self-aware computing system is the target system extended with reflexivity, in a reflexive system there is a subtle but important change in the nature of the system. Specifically, the task performed by the target system becomes one of a possible set of behaviors of an agent provided with attributional self-identity and, possibly, a higher level of control. In other words, what we call “the system” is no longer the target extended with reflexivity, but the reflexive agent provided with the target process as one of its possible behaviors.

Computational Subjectivity
Dennet (1991) describes a process by which organisms and minds develop self-identity. He metaphorically defines the mechanism of self-development as something produced by a “center of narrative gravity.” It can be seen as an inner storytelling by which the organism tells its own story. It creates its self-identity and re-organizes itself according to it. Dennett also suggests that we are the product rather than

These considerations, when applied to a system provided with computational reflexivity, can be interpreted by claiming that the self-model might not necessarily be a full representational model of the target system but, instead, be limited to acquire an attributional function. When a system becomes capable of saying “This is me”, then it can connect actions and perceptions as its own. Thus, it can produce genuinely subjective content that can be accessed and reported as subjective reports.

Here we define self-spinning in a technical and not metaphorical way, as a self-organizing structure (the subject) responding to actions or perceptions and attributing them to itself at a meta-level of representation. An advantage of identifying the subject as an attributional entity, instead of a fully representational one is that it does not require a large amount of common-sense knowledge (Mitchell, 2019). In this case, the focus is not on the development of a complex model of personal identity, but on providing the capability to assign a given perception or action to a prefixed “self” as a “center of narrative gravity.” We emphasize that, as a specific type of computational reflexivity, the execution of the target task and self spinning (or subject formation) are two processes 1) occurring concurrently and 2) mutually influencing, in such a way a bottom-up self-organizing process is merged with a top-down feedback one (Carver and Scheier, 2002).

Finally, we can define computational subjectivity as the capability to produce dynamic subjective content, which is in turn defined as a temporal list of structures pairing the object (i.e., the current execution of the target task) and the subject (as defined above). In the next section, the notion of subjective content is expressed as a specific knowledge structure called subjective arc.

System Description

The prototype has been designed according to the ideas introduced above. It is composed of three main subsystems: the Poetic Line Selector, the Trait Clustering System, and the Subjective Arc Generator. This section provides a brief description of each of them and then an outline of how it works.

The building block of the system is a function for the measurement of semantic similarity between words of the English lexicon. In the context of natural language processing, word embedding is a class of techniques for representing words and documents as vectors. In particular, a number of metrics have been defined to measure the distance between the words in the vector space (or word similarity). The most common of them is the cosine similarity. When the word similarity is high (and, correspondingly, the vector distance is low), the words show a strong association in the common-sense knowledge.

To measure word similarities we employ word embedding provided by Spacy\(^1\), an open-source software library in Python for advanced natural language processing (Hippopata, 2021; Jurafsky and Martin, 2000). In particular, we use word2vec model for word embedding and semantic similarity: Jatnika, Bijaksana, and Suryani (2019) trained it on a large-scale language model in English\(^2\).

The Object: Poetic Line Selector

To have a simple target system, we implemented a procedure for generation of text through the selection and collection of poetic lines according to prefixed semantic dimensions. Although it would be too ambitious to call it a “poem generator”, the system has the advantage, on one hand, of being easily tuned with a few parameters, yet, on the other hand, enough complex to generate texts with some degree of semantic consistency. As a next step of this research, the system could be used to empirically compare the overall aesthetic perception of the target system alone and its counterpart provided with computational subjectivity.

According to a prefixed text length (i.e., the number of lines to be extracted), the procedure iterates on the selection of one line at a time from the Gutenberg Poetry Corpus\(^3\) (Jacobs, 2018), a large collection of poems. Once randomly picked up, the candidate lines are converted into a list of content words and filtered according to the semantic similarity with two input keywords. A minimum value is prefixed for the similarity value. Although there is no limitation in the choice of the two keywords, we use the first keyword as a topic word (e.g., holiday) and the second one as an emotion word (e.g., love). So, they provide two different semantic dimensions according to which the list of lines shows, to some extent, semantic consistency.

Valitutti, Strapparava, and Stock (2008) implemented an analogous form of “semantic slanting”, where a topic word and emotion word was employed to generate advertising headlines. A more sophisticated approach to slogan creation was reported by Alnajjar and Toivonen (2020). A few example lines generated by our prototype and corresponding to the topic word ‘soul’ and the emotion word ‘love’ are shown in Table 1.

The Subject: Trait Clustering System

To provide a “subjective augmentation” of the target task, we need a procedure building something recognizable as a ‘self-identity’ to be associated to the target process in runtime. To this aim, we built a collection of 205 adjectives denoting personality traits and then implemented a simple clustering system based on the semantic similarity metric described above.

At the first run, if we provide a word in input, the procedure selects the most similar trait adjective and put it as a single cluster (i.e., a list containing that word). As long as

---

1https://spacy.io
2spacy.io/models/en/en_core_web_lg
Overall, the corpus contains more than 3 million poetic lines.
we add new words as input, the procedure will either create new clusters or add the word to an existing cluster. The choice is based on a prefixed value for the minimum similarity. If the similarity with the most similar adjective is above this threshold, a new cluster is created. Table 2 shows a subset of clusters generated using 0.45 as the value of similarity threshold.

Since the trait clustering should be proceed concurrently with the target task and be informed by it, we defined a way to pick up a word pair containing a content word and a trait word from the current poetic line. More specifically, once a poetic line is chosen according to the method described above, one of its content words is further selected. As a ranking function we define a combination of similarity between the words paired to the input keywords and each trait adjective. If we call \( S_1 \) the first similarity value and \( S_2 \) the second one, then the total similarity \( S_{TOT} \) is obtained by this relation:

\[
S_{TOT} = \frac{S_1 + S_2}{2(S_1 + S_2)^2 + 1}
\]

According to it, an adjective word (called trait word) is selected. In summary, each line generates a trait word which, in turn, triggers a new clustering step.

---

Table 2: A selection of trait clusters generated using 0.45 as minimum threshold for semantic similarity.

<table>
<thead>
<tr>
<th>Line</th>
<th>Word Pair 1</th>
<th>Word Pair 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Life was in him so passing beautiful!</td>
<td>(life, soul) → 0.584</td>
<td>(beautiful, love) → 0.595</td>
</tr>
<tr>
<td>And the joy of the meetin’ bethuxt him and me</td>
<td>(joy, soul) → 0.598</td>
<td>(joy, love) → 0.628</td>
</tr>
<tr>
<td>Our hearts, our hopes, are all with thee,</td>
<td>(heart, soul) → 0.573</td>
<td>(hope, love) → 0.693</td>
</tr>
<tr>
<td>One ear fulfilled and mad with music, one</td>
<td>(music, soul) → 0.505</td>
<td>(mad, love) → 0.502</td>
</tr>
<tr>
<td>The vain and passionate longing came again</td>
<td>(longing, soul) → 0.596</td>
<td>(passionate, love) → 0.511</td>
</tr>
</tbody>
</table>

---

The **Subjective Content: Subjective Arc Generator**

The set of trait clusters is not yet the “subject” but a set of “potential selves”. The system needs to either select a new cluster as the current subject or confirm the one previously selected. In this case, the decision-making process takes into account two elements: 1) the similarity value between the current line and the newly selected trait word, and 2) the decay value associated with the current subject. The decay value is the product of the previous decay value (initialized to 1 each time the subject is changed and, thus, associated to a new trait cluster) and a prefixed decay factor. In this way, we can provide a proper weight to the duration of a given subject. If it is meant to simulate an emotion, the decay factor should be small. On the other hand, if it is designed to simulate a more stable mood or personality, it should be closer to 1. The employment of a decay value was also inspired by the time decay function used in the Reddit ranking algorithm (Stoddard, 2015). Table 3 shows a sequence of steps in which the Subject is confirmed or changed according to the trait similarity and the decay value.

Once selected the current subject from the trait cluster, a new poetic line is checked according to it. In other words, to be part of the “subjective node”, the line should have a sufficient semantic similarity not only with the input keywords but also with the subject, in such a way to be informed by it. At that point, both the bottom-up and the top-down aspects of the iteration are completed. In each iteration instance, the system picks up the selected line and the current subject as a pair expressing the current subjective node.

Table 4 shows the main steps leading from the two input keywords (‘robbery’, ‘regret’) until the generation of the subjective node. In this example, we see that the line initially chosen to generate the trait word for updating the clusters may differ from the line selected at the next stage, as the most similar one to the current subject. This distinction emphasizes that what is recorded as the object of the subjective content is not the initial action that lets the system update the subject, but instead the action executed according to the current subject. This information makes the system potentially capable of generating verbal reports such as “I did this because I am that way”. The subjective arc, that is the list of subjective nodes built along with the iterations, is the final and central output of the system: a knowledge structure expressing something recognizable, to some extent, as a ‘flow of consciousness’.

More prosaically, what we have here is a content that
<table>
<thead>
<tr>
<th>N</th>
<th>Line</th>
<th>Subject</th>
<th>Trait Similarity</th>
<th>Decay Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Low lies the heart that swell’d with honest pride.</td>
<td>Passionate</td>
<td>0.606</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Where I, to heart’s desire,</td>
<td>Passionate</td>
<td>0.592</td>
<td>0.8</td>
</tr>
<tr>
<td>3</td>
<td>My light in darkness! and my life in death!</td>
<td>Evil</td>
<td>0.592</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>When the heart grows weary, all things seem dreary;</td>
<td>Just</td>
<td>0.764</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>Or human love or hate;</td>
<td>Just</td>
<td>0.584</td>
<td>0.8</td>
</tr>
<tr>
<td>6</td>
<td>The heart for which she cast away her own;</td>
<td>Passionate</td>
<td>0.461</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3: Sequence of 6 next iterations. The input keywords are `blood` and `fear`. The head `Subject` is represented here by the first word in the currently trait cluster. In some cases, the change of the Subject is due to the reduction in the decay value. The decay factor here is 0.8.

<table>
<thead>
<tr>
<th>Step</th>
<th>Variable Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>(Input)</strong>&lt;br&gt;word1 = 'robery'&lt;br&gt;word2 = 'regret'&lt;br&gt;min_similarity = 0.45&lt;br&gt;trait_clusters = trait_clusters_045</td>
</tr>
<tr>
<td>2</td>
<td><strong>(Candidate Line Selection)</strong>&lt;br&gt;line = From chains and prisons, ay, from horrid fear.&lt;br&gt;word_info1 = ['robery', 'prison', 0.486]&lt;br&gt;word_info2 = ['regret', 'fear', 0.609]</td>
</tr>
<tr>
<td>3</td>
<td><strong>(Subject)</strong>&lt;br&gt;trait_info = ['fear', 'evil', 0.608]</td>
</tr>
<tr>
<td>4</td>
<td>selected_cluster = ['evil', 'vain']</td>
</tr>
<tr>
<td>5</td>
<td><strong>(Object)</strong>&lt;br&gt;informed_line = And let the worst thou yet hast done be innocent.</td>
</tr>
</tbody>
</table>
| 6    | **(Subjective Node)**<br>("And let the worst thou yet hast done be innocent",
                             ['evil', 'vain']) |

Table 4: Example sequence of steps from the input couple of keywords to the generation of the subjective node.

As future work, we intend to use the system as a testbed for performing empirical evaluations with human users. Specifically, we aim to test if the introduction of computational subjectivity, as defined here, can increase the perceived creativity of the selected text in a statistically significant way. Proper tuning of the parameters such as minimum similarity and decay factor can help us to test different levels of granularity in the clustering process and to study the effect of different time duration of the current subject. A possible next integration with well-known models of personality such as the Big Five Model could be performed to explore the connection between self-identities and the lexicon referred to them.

A further step would be the application of computational subjectivity to different and more complex creative tasks such as story generators, music composers, or adaptive chatbots. A second aimed line of future research would be focusing on a single complex task such as narrative generation, and using available state-of-the-art systems such as MEXICA (Pérez y Pérez and Sharples, 2001) or Scéalextric (Veale, 2016) as targets. In this case, different versions of the system could implement the subject as either the author of the narrative or a fictional character.

Finally, the generated subjective arcs would allow the system to produce and integrate the subjective point of view into the narrative and, thus, provide the second-order dimension worth to be assessed.

As future work, we intend to use the system as a testbed for performing empirical evaluations with human users. Specifically, we aim to test if the introduction of computational subjectivity, as defined here, can increase the perceived creativity of the selected text in a statistically significant way. Proper tuning of the parameters such as minimum similarity and decay factor can help us to test different levels of granularity in the clustering process and to study the effect of different time duration of the current subject. A possible next integration with well-known models of personality such as the Big Five Model could be performed to explore the connection between self-identities and the lexicon referred to them.

A further step would be the application of computational subjectivity to different and more complex creative tasks such as story generators, music composers, or adaptive chatbots. A second aimed line of future research would be focusing on a single complex task such as narrative generation, and using available state-of-the-art systems such as MEXICA (Pérez y Pérez and Sharples, 2001) or Scéalextric (Veale, 2016) as targets. In this case, different versions of the system could implement the subject as either the author of the narrative or a fictional character.

Finally, the generated subjective arcs would allow the system to produce and integrate the subjective point of view into the narrative and, thus, provide the second-order dimension worth to be assessed.
Abstract

Most AI-based dance generation methods consider dance as a selection of movements. While these methods aim for realistic outputs and synchronization with music, domain-specific knowledge related to dance is often not incorporated in these models. This results in outputs that are far from what the dance community considers as high quality dances. Therefore we present CoPE, a dance education framework for AI systems with three key components: Composition, Performance, and Evaluation. By analyzing nowadays dance education, we suggest how to incorporate principles from the dance community into AI-based dance generation systems by aiming for more realistic pieces of art.

Introduction

Over the past few years, the development of generative models and pose estimation methods has shown major improvements (Goodfellow et al. 2014), (Güler, Neverova, and Kokkinos 2018), (Cao et al. 2019). These methods have been further developed for AI-based dance generation methods while aiming for high quality outputs, realistic body movements, a large variety of dance styles, and a proper synchronization between music and dance (Chan et al. 2019), (Li et al. 2020). For the majority of these works the goal is either motion transfer from experienced dancers to inexperienced dancers by using video-to-video translation or to create a sequence of new dance movements. In this light, dance is considered as a selection of movements, at times performed on music.

The dance community considers dance as more than a selection of movements. Dance is an art form that aims to communicate an idea through movement, which must have a foundation for logical development (Smith-Autard 2010). To compose a dance is to create a work of art. According to (Robinson 2009) ‘it involves putting your imagination to work to make something new, to come up with new solutions to problems’. (Redfern 1973) describes that ‘dance can be considered as art when it is formulated as a whole or a structured something, where the relationship and coherence of constituent parts increases the interest and importance’. A dance that is created with the composer’s intention to communicate an idea or emotion, can be for example about people, happenings, moods, or even the movement itself (Smith-Autard 2010). Movement content is selected as an abstraction of the actual feeling or happenings to suggest meanings that are significant to the dance idea.

By comparing AI-based dance generation methods and composers or choreographers, we can distinguish a clear difference in their creation processes. Dance generation methods take existing dance videos and let the model learn from the movements that are present in the video. Note that all dance generation methods considered in this work are AI-based, but we use the terms interchangeably. The movements, whether copied or inspired by the original dance videos, are then evaluated by metrics that measure the similarity between movements and sometimes their alignment with music. The dance community has used several methods to educate dance over the twentieth century. Back in the early days, it was focused on a Laban model which emphasized the experimental child centered process of dancing as a means of developing personal qualities. Then it shifted towards a theater art model with an emphasis on the dance product. More recently (Smith-Autard 2010) introduced a midway model, one that incorporates aspects of both the Laban model and the theater model. This method has been widely used at universities for performing arts. She distinguishes three key components that are central in nowadays dance education for students: composing, performing and appreciating/evaluating dances.

Previous works have mainly explored how practices from dance composition can be used in AI systems. In (Jacob et al. 2013), researchers in computational creativity collaborated with a contemporary theater group and created interactive narrative artworks with the help of an agent, considering perception, reasoning and action modules. They used a compositional technique named ‘viewpoints’ that enables dancers to communicate through movement and gesture, builds a vocabulary and allows them to improvise. (Rett and Dias 2017) studied to what extent Laban’s Movement Analysis can be implemented with the help of probability calculus and Bayesian theory. Also (Gujrania, Long, and Magerko 2019) studied Laban’s Movement Analysis and presented a framework that can be applied to real-time interaction in games by using full-body gestures. These studies have a main focus on dance composition and less on the performance and evaluation aspects. Furthermore the type of AI systems considered in these works are mostly interactive and agent-based whereas recent studies on AI-based dance
generation methods use Deep Learning based methods. In this work, we study to what extent dance generation methods can learn from nowadays dance education. We present the framework CoPE that is based on the key components Composition, Performance, and Evaluation, which can be used by dance generation methods. For each of these three components we describe its role in dance education, its main components, and suggestions to incorporate it into dance generation methods. Our aim is not to cover all aspects of dance education, but to present a starting point. For developing dance generative methods we encourage researchers in computational creativity to use dance education handbooks such as (Smith-Autard 2010) or to involve dance professionals to include the dance-specific domain knowledge into new dance generation methods.

Composition
The composition of a successful dance requires that the composer has knowledge of 1) the material elements of a dance, 2) the methods of constructions which give form to a dance, and 3) an understanding of the style in which the composer is working (Smith-Autard 2010). No book can provide all the ingredients necessary for the production of dance art works. For students in dance, it is up to the student’s imaginative use of the concepts and principles that should be promoted through a program designed by a dance teacher. Other useful resources are attending live performances, recordings of professional choreography and workshops from professional dance artists. This section summarizes the three principles for composing a dance that are defined by (Smith-Autard 2010). For each of these principles, recommendations for dance generation methods are provided.

The Material Elements of a Dance
Dance Education In daily life, communication can take place through movement. Many verbal expressions describe moods or thoughts in terms of movements, for example ‘jumping for joy’. Since dance is used to communicate an idea, it can be seen as a language. While dance composition might use every day movements, they are usually used as an abstraction of the actual feeling or message. The material elements of a dance have been described by von Laban and are still widely used today (von Laban and Ullmann 1971). He broke down dance into four main components: action of the body, qualities of movement, space environment, and relationships. The action of the body considers the body as an instrument that can use body parts and make body shapes. The quality of movement is related to properties of time, weight and flow, where combinations of these aspects can determine the form and style of an action, e.g. ‘quick’ and ‘light’. The space environment describes how the body moves into space, by considering the size of the movement, pathways on e.g. the floor and in the air. Lastly, the relationship describes how the dancer relates to objects and other people in the space and how movement responds to this, e.g. by mirroring movements or dancing in canon.

Dance Generation Methods The action of the body itself has the main focus and the other three components defined by the Laban method are less studied. For example, different movement qualities might be learned from existing seen dances unconsciously and used randomly, but methods don’t derive those actively. Regarding space and relationships, most methods have used training data where a single dancer is dancing with a focus to the front, without any interaction with objects or other dancers and without exploring the space. Enforcing the other material elements of dance can be done by using a wider variety of input videos for training models, that are not focused on a single dance performer in front of a camera, e.g. recordings of dances in the theater performed by multiple dancers. Another way is to set constraints to the model related to moving in space or interactions between dancers or a dancer and an object. Letting the generative model independently learn how to select the best movement quality for a particular movement in a sequence is a challenging task. An easier way to incorporate movement quality in generated dances would be to use manipulative generative methods with the use of attributes, to be able to adapt the dance after it has been created.

The Methods of Construction
Dance Education The beginnings of composition consist of stimuli, decision on the type of dance and the mode of presentation, improvisation, evaluation of improvisation, selection and refinement, and motif (Smith-Autard 2010).

Stimuli for dance compositions can be auditory (e.g. music, words), visual (e.g. pictures, objects, patterns), ideational (ideas), tactile (e.g. the feel of a material) or kinesesthetic (movement itself). Whenever one or more stimuli are selected by the composer, the type of dance is selected. Types of dances can be described by styles such as ballet, modern, jazz, contemporary, hip-hop etc. or by mode of presentation, such as pure, abstract, lyrical, or dramatic. Nowadays the division between styles is less strict and more and more choreographers use combinations.

After deciding on the type of dance and mode of presentation, a composer starts experimenting with movements and tries to realize imagined movement images into real movement expression. This initial exploration is called improvisation. In this phase, the composer may use several criteria to decide whether movements are a suitable starting point of this work: 1) the movement has meaning or relevance to the idea for the dance, 2) the movement is interesting and original in action, dynamics and spatial patterning, 3) the movement has potential for development. Whenever the starting point for movement is selected, evaluated and refined, it can be set as the initial motivating force for the rest of the dance. This movement or movement phrase is called the motif.

When the motif is established, the composer will create dance material and arrange this. The motif can be repeated by varying aspects related to the material elements of a dance described before: action, quality, space, and relationship. However, the goal of the dance is to have a form and an overall shape, since it aims to communicate an idea and therefore is much more to it than an arrangement of movements. For a viewer who has seen the dance, the goal is not to remember each movement and the order of the piece. Rather, the impression of the whole is remembered,
its shape, the excitement of the development, its main message and its originality. So apart from selecting movement content, the composer should set the movement into a constructional frame which will give the whole its form.

Dance Generation Methods  The main focus is on kinesesthetic movement as a starting point, because of the use of dance videos as training input. Sometimes auditory input can play a part, such as in (Tendulkar et al. 2020), although in most works music is only used for alignment with the dance. Other stimuli are not considered, but could be extracted from non-dancing videos that represent those stimuli and can be shown to the model in the training phase, for example by letting the model extract shapes from natural movements caused by waves or the wind. Another approach could be to create methods that combine different multimodal inputs, to represent e.g. shapes, ideas or patterns, such as audio or single images.

Apart from motifs, (Smith-Autard 2010) distinguishes seven other important elements of construction: repetition, variation and contrasts, climax and highlights, proportion and balance, transition, logical development, and unity, where unity is the overall aim. In dance generation methods, repetition of dance movements on specific parts of the music is sometimes considered and occasionally climaxes are also linked to music. However, all other aspects are not the key of attention and not integrated intentionally. Some of them are not easy to integrate, but important to create enough interestingness for the dance. As an example, transitions in dance are highly important and are among the most difficult aspects of dance composition (Smith-Autard 2010). Transitions can be between positions or between sections of the dance. Nonetheless, in dance generation the focus is not on transitions, but on dance movements themselves. (Lee et al. 2020) take a first step in this by disentangling a dance unit into two latent spaces: an initial pose space and a movement space. An explicit consideration of all these elements might be very useful to create realistic dances that are interesting to watch.

The Style of a Composer

Dance Education  Style in dance is considered the same as technique, since it often means the content of the idiom and not only how it is presented (Smith-Autard 2010). Each technique consists of a set of principles that need to be mastered to perform a dance piece in the way it is created. To master this technique, a dancer needs years of training to acquire these skills, to exercise the body and to practice movements to achieve perfection. For example ballet requires turnout, which is a rotation of the leg at the hips which causes the feet and knees to turn outward. Popping is a dance style that requires extreme control in contracting and relaxing body muscles, e.g. in the arms and chest. On top of this style, each choreographer has its own personal style, which is a personal interpretation and a way of using the technique.

Dance Generation Methods  Different styles have been considered, varying from ballet to hiphop and commercial styles, although the majority is focused on the latter, since its ease of use of having a single dancer facing the front. However, current methods do not use any style specific information regarding rules or constraints. This results in generated dances that are full of incorrectly performed movements when considered on a technical level. For example, ballet dances are created where representations of bodies did not learn the exact posture of the body that is required for this technique. When specific styles such as popping are given as input data, models are able to copy the shapes of the movement, but do not learn the principle of muscle contraction. This is caused by the limited representation of the human body by the model that only consists of a set of keypoints and does not represent muscles. We therefore encourage researchers in this field to focus on a specific style or set of similar styles as a starting point.

Performance

Dance Education  Performing a dance on stage differs from dancing in a classroom or studio. Dancers strive to perfection for this moment and aspects such as a decor, light, and costumes will be added to make the performance a whole. Apart from mastering the physical and technical skills, it is important to bring the dance with the right intention and emotions as suggested by the composer. (Smith-Autard 2010) describes that traditional approaches to learn a dance piece were very focused on copying the teacher or dance artist who performs the piece first. These days dance students are often involved in the creation process and encouraged to involve their own style and personality in the performance. During the process, dancers are corrected to come as close as possible to the perfect execution of the dance that the choreographer has in mind.

Dance Generation Methods  As explained before, the final product aims to match an existing dance as close as possible or to create new movements. It is up to the model to correct dance movements in some way and to make sure the output is as desired. But in the end, the output is not as optimal as we would like, since as discussed in the previous section, not all items that should be taken into account are considered. Instead of putting constraints from the start, an option would be to incorporate user feedback from experts, either during the training process or by being able to make modifications to the final outcome.

Another aspect that contributes to a correct execution of the movement is the representation of the body. The representation of the human body in joints in dance generation methods varies from just a basic set of key points as shoulders, elbows, hands, knees, feet and hands by (Zhuang et al. 2020b), to 14 slightly more detailed keypoints by (Lee et al. 2020) and 63-dimensional 3D joint positions by (Ahn et al. 2020). However, the use of specific constraints for each style with respect to keypoints, rotations and muscles, as explained before, will be required to create realistic performances.

Evaluation

Dance Education  According to (Smith-Autard 2010), there exists no objective formula to evaluate a dance. It can
only be measured as successful relative to the experience and background of the person who sees it and to the composer’s stage of development in composing. However, there will always be aesthetic judgments. Although there is no universal set of criteria to compare dances, one of the environments that uses criteria are commercial dance competitions. Here the judging is typically a guided rubric that evaluates performance, choreography, technique, and overall appearance. More specifically judges tend to look for quality technique, performance quality, clear lines and transitions, musicality, personal style, professional overall appearance, appropriate costume choices, and memorability (Wollins 2014). However, these criteria are not standardized.

Dance Generation Methods Mostly quantitative evaluation methods have been used and sometimes user studies with non-experts in dance. Regarding metrics, the Fréchet Inception Distance (FID) is used to measure how close the distribution of generated dances is to the real (Huang et al. 2020), (Lee et al. 2020), (Zhuang et al. 2020a), (Zhuang et al. 2020b) and the correspondence between each dance step to a temporal window in the input music is measured by a Pearson correlation (Tendulkar et al. 2020). Some works (Huang et al. 2020), (Tendulkar et al. 2020) compare dances in user studies based on musicality, unpredictability, creativity, inspiration, realism, smoothness, and matching with music. (Zhuang et al. 2020b) uses the same criteria for generated dances individually without making a comparison. On the other hand, (Pettee et al. 2019) evaluates its algorithm by letting it assist choreographers during a creation process and by analyzing the latent space.

Evaluating dance is subjective to some extent, but aspects related to the dance technique can be judged in an objective way. Expertise in evaluating dances comes with practical dance experience as well as watching many professional dancers and companies. Whenever dance generation methods will be evaluated by non-experts in dance, we might end up optimizing for either simplified or even incorrect metrics. Therefore, this field would highly benefit from involving dance professionals in this evaluation process.

Discussion

Most recent dance generation methods aim to transfer dance movements to inexperienced bodies or aim to create new choreographies. However, there is a large gap between dance generation methods and dance education practices. For this reason, we present the CoPE framework to give guidelines for developing realistic dances by generative models. In general we would highly encourage researchers in computational creativity to work together with dance practitioners to get a better feeling of the complexity of creating a dance and to ensure something realistic is created. An example of such a successful work is (Pettee et al. 2019), which is a multi-disciplinary project done by a team of dance artists, physicists, and machine learning researchers. By working closely together, they show that technology can actually help stimulating creativity in the dance composition process. By encouraging such collaborations, we hope that future dance generation methods will be on such a level where they are able to inspire choreographers and help them in their composition process.

References


Jacob, M.; Coisne, G.; Gupta, A.; Sysoev, I.; Verma, G. G.; and Magerko, B. 2013. Viewpoints AI. In *AIIDE*.


Towards a Visual Language Using Neural Networks

Luís Gonçalo, João M. Cunha, and Penousal Machado
University of Coimbra, CISUC, Department of Informatics Engineering
{lgoncalo,jmacunha,machado}@dei.uc.pt

Abstract

In recent years computer simulations have proven to be useful in the study of the origin and evolution of communication. In this paper, we present a system that is able to evolve image-based communication protocols to transmit information. We trained an encoder and a decoder in an architecture similar to a model of communication where the generator transforms a message into an image and the decoder tries to reconstruct the original message. This way the two networks are stimulated to work together to establish some type of communication.

We used the GloVe word to vector dataset to generate images for concepts and took advantage of its linear properties to generate new concepts. We analyse the results by comparing images of similar concepts and demonstrated that our system is capable of creating similar images for related concepts, distancing itself from different concepts.

Introduction

Even before the existence of a formal writing system, the human species developed ways to communicate knowledge using proto-writing which consisted of ideographic symbols that represented a limited number of concepts (Schmandt-Besserat, 2014).

In recent years, there have been proposals to design a universal written language based on ideograms. One of the examples is a system of Blissymbols proposed by (Bliss, 1965) which was conceived as a writing system where each basic symbol represents a concept and can be combined to represent new concepts. There also have been many approaches to study the origin and the evolution of language using computer simulations. However, the majority of these approaches are focused on the evolution of communication based on symbols (Sukhbaatar, szlam, and Fergus, 2016; Forerster et al., 2016; Mordatch and Abbeel, 2018) or textual communication (Das et al., 2017; Lewis et al., 2017).

In this paper, we propose a system that generates images that represent single concepts. We trained two neural networks to create a vocabulary composed of concept representative images through collaboration. In addition, we present a thorough analysis of the results obtained taking into account the word to vector linear properties.

Related work

There have been recent approaches to generate concept-representative symbols in the field of Computational Creativity. One example, the Emojinating system (Cunha et al., 2020) which generates visual representations of concepts through visual blending emoji. Xiao and Linkola (2015) also presented a similar system based on the combination of images. This system takes a conceptual task described in text and generates visual compositions taking into account the semantic associations and using different visual combination operations.

However, in all of these approaches the initial input, already exist and the system only generates a combination of them. Some approaches try to tackle these limitations by using machine learning techniques to generate images such as Generative Adversarial Networks (GAN) (Goodfellow et al., 2014).

These networks have been used to simulate the model of communication proposed by Shannon (1948). The author states that the general communication system can be divided into five elements, the information source which produces the message, the transmitter which creates the signal, the channel which is the medium that carries the information, the receiver transforms the signal received to the original message and the destination for which the message is intended.

Simon¹, Trenaman² and Murdock³ used this model of communication together with GAN to develop a small set of experiments inspired by early proto-writing systems such as cuneiform and hieroglyphs. In these approaches, the encoder turns a message into an image that the decoder tries to decode back into the original message. Noise is added in the communication channel and increased over time so the encoder and decoder are encouraged to evolve the language to be robust to noise. Park (2020) presents a similar approach but focused on symbols. In this approach, the encoder is enforced to come up with a set of distinctive symbols that

¹https://www.joelsimon.net/dimensions-of-dialogue.html, retr. 2021
³https://rynmurdock.github.io/2020/02/05/CCN.html, retr. 2021
resemble the human-made glyphs. The authors adopted a GAN-based neural painter trained with a synthetic brush-stroke dataset so the encoder focuses on generating a set of symbols with diverse shapes that resemble human-made glyphs.

**Approach**

Some approaches use this type of model combined with neural networks to evolve a visual language, however, in our approach, we not only evolve a visual language, but each artefact that is transferred between networks has an underlying concept associated. However, our system aims to generate images that visually represent concepts by jointly training an encoder and a decoder to transfer a representation of a word through a noisy channel, following the Shannon model of communication (Shannon, 1948).

To obtain a vector representation of concepts, we used a word to vector dataset called Global Vectors for Word Representation (GloVe) (Pennington, Socher, and Manning, 2014) These representations showcase interesting linear substructures of the word vector space. For example, the underlying concept that distinguishes man from woman may be equivalently specified by various other word pairs, such as king and queen or brother and sister.

For the generation of the message to be transmitted, which in this case is an image, we used two neural networks based on the Deep Convolutional Generative Adversarial Network architecture (Radford, Metz, and Chintala, 2016), provided by Pytorch. To implement the architecture proposed by Shannon (1948), we modified the decoder implementation so its output is a reconstruction of the input vector of the encoder. The training process is similar to the one used in autoencoders, however as the communication channel is noisy the signal received by the decoder differs from the original one.

First, the encoder tries to encode a set of word vectors from the GloVe dataset in RGB or grayscale images. Then, some type of noise is applied to the generated images based on a set of transformations. These transformations consist of a set of rotations, translations and normalization of the pixel data. Finally, the decoder tries to reconstruct the original vectors based on the images received.

The quality of the encoder and the decoder is assessed by evaluating how well the decoder is able to reconstruct the original vector. This way, the two networks are forced to cooperate to be able to converge to a vocabulary that is understandable by both.

**Experimental setup**

In this section, we describe the setup used in our experimentation. As previously mentioned, the encoder network is based on the original implementation of the DCGAN with some modifications on the last layers of the network.

The loss value is calculated using the mean squared error function which measures the average squared difference between the estimated values and the actual value. In our case,

\[ \text{loss} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \]

it measures the difference between the reconstructed latent vector \( z \) and the original vector used in the encoder. The networks were trained during 50 epochs. The batch size used was 64 and the learning rate was set to 0.001 which was divided by 10 at epoch 25, 35 and 43. Finally, the vector size used to set the size of the input layer of the encoder and the output layer of the decoder was set to 100 to be the same size as the word vectors of the dataset.

**Visualization Tool**

To more easily explore the results obtained we developed a webpage that presents the generated images. We used a t-distributed stochastic neighbour embedding (van der Maaten and Hinton, 2008) to transformed our higher dimensional vectors into a representation that we can visualize, in our case two dimensions to use as \( x \) and \( y \) value and created a 2D world by placing the images in the corresponding positions. Figure 1 shows a screenshot of the developed website.

On the website, it is also presented a version in which the images were created using three channels (red, green and blue). The website also provides a search tool to more easily find the images for each concept and compare them with other images and a different distribution based on the similarities between the images instead of the word vectors.

**Results and discussion**

Using the training process described in the previous section, we trained the networks to generate images for concepts taken from the GloVe dataset. Firstly, we removed the English stop words and words that may not have any concepts by themselves, such as ‘he’, ‘she’ and ‘it’. In the end, we choose the 5120 most popular words of the dataset.

**Analysis of similar and distinct words**

Usually, the quality of autoencoders Makhzani et al. (2016) can be analysed based on the similarity of the vectors produced for each image. If the original images have strong similarities, the vectors produced must have similarities.

---

4[https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html](https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html)

However, when two images are different the autoencoder must be able to create distant vectors for each image. So, we decided to compare images of similar and different concepts to infer if the model is capable of creating similar images for related concepts while distancing them from different concepts.

Figure 2 presents two pairs of images, a similar pair (‘man’ and ‘woman’) and a different pair (‘islam’ and ‘chrysler’). As it is possible to observe, the two images that represent the words ‘man’ and ‘woman’ present very similar characteristics. Even though they are two antonyms, the context where they emerge is similar which results in similar images. The word to vector training is focused on the word associations, not on the meaning of the words, therefore, the word vectors that are closer are the ones that emerge in similar contexts rather than similar words. So, it is expected to find significant similarities between the images that represent ‘man’ and ‘woman’. The second pair (‘islam’ and ‘chrysler’) was selected based on the two words from the dataset with the biggest distance between them. As it is possible to observe, the two images are very different from each other, which is expected as they have very different word vectors. This shows that our model can emerge a vocabulary that approximate images that represent similar concepts while distancing images for concepts that are not related.

**Analysis based on vector operations**

One of the properties of a word to vector architecture is the linear substructures of the word vector space which can capture multiple different degrees of similarity between words. Mikolov, Yih, and Zweig (2013) found that semantic and syntactic patterns can be reproduced using vector arithmetic. Patterns such as man are to woman as king is to queen can be generated through algebraic operations on the vector representations of the words. In this analysis, we investigated if the semantic and syntactic patterns of the word to vec-

tor can be reproduced in our model by obtaining vectors of concepts that should be similar to existing concepts in the dataset. Then, we produced the images for both vectors and compare them visually. This way, we can infer the quality of the model on the generation of concepts that may not be available in the original dataset.

In the first experiment, we calculated the distance from the vector that represents the word ‘china’ to the vector that represents the word ‘chinese’, which represents the distance that goes from the country to the inhabitant in that country. Then, we created another word vector by adding this distance to a vector that represents a different country and generated the corresponding image. We used the following formula to synthetically create the vector for the word ‘american’ (referred to as ‘interpolated american’) and Figure 3 presents the results obtained.

\[
\text{chinese} - \text{china} + \text{america} = \text{american}
\]

As it is possible to observe in Figure 3 the image generated using the real vector (‘american’) and the image generated using the word vector created using the method previously described (‘interpolated american’) are very similar, which indicates that our model can produce images similar to ones of existing vectors through vector operations. This can be used to produce images for concepts that are not available in the dataset.

We expanded our analysis beyond countries and nationalities to assess if these properties can also be observable in verbal tenses. First, we calculated the distance that goes from a verb in the present tense to its past tense, for example from ‘go’ to ‘went’, and synthetically created past tenses for other verbs.

Figure 4 presents the comparison between both images, the image generated using the vector that we synthetically created and the image generated by the original vector that represents the word ‘took’. As it is possible to observe, our model is also able to generate images to represent the past tenses using the present tense. This might be useful when the past tense of a verb is not available but the present tense is, so we can generate the word vector synthetically and then the image that represents it.

**Analysis based on the visualization tool**

With the help of the visualisation tool, it is possible to observe some behaviours that can be unexpected as some con-
concepts may have similar images while for us they may not be related. One example of this behaviour is related to the word ‘one’. In our visualisation tool, it is possible to observe some groups of words that are formed with related words, for example, the group related to sports, where words such as ‘football’, ‘team’, ‘player’ are located separated from the rest. The same behaviour is observed in the numbers. However, the word ‘one’ is not in the same place as the other numbers. As the word ‘one’ emerges more associated as a single unit or individual, like the word ‘only’ or ‘another’. So, as it is more used in these contexts, the work ‘one’ is placed farther from the other numerals.

Conclusion and Future Work

Over the past years, we observed the adoption of computer models and artificial intelligence on language evolution. In our approach, we used two neural networks to evolve communication protocols based on images where each image represents a different concept, where similar concepts lead to images with similarities. We also explored the linear properties of the word vector by using vector arithmetic to create images for concepts. This approach proved to be useful to generate concepts that the dataset does not contain or even new non-existing concepts using the available concepts.

One of the limitations of the developed approach is the low resolution of the generated images. As the output size of the neural network is 64x64 pixels. Also, the images could be adapted to use graphic objects instead of raw pixel data. This way the images would be scalable without loss of quality and could be adapted to create a writing system more similar to the one we use.

It is also important to highlight the potential of our approach; it evolves a communication medium based on the visualization of a word embedding. It can be used in the generation of visually aesthetic QR codes which can only be deciphered using the correct decoder. It is also possible to add constraints to the generated images to enforce the emergence of customizable characteristics, such as controlling the amount of the black or white colour in each image.

However, this approach is focused on computer communication that humans can observe and analyse. It represents the first step toward the creation of systems capable of evolving language, where we can use abstractions from visual features of the world.

References


Mordatch, I., and Abbeel, P. 2018. Emergence of grounded compositional language in multi-agent populations. In AAAI.


Sukhbaatar, S.; szlam, a.; and Fergus, R. 2016. Learning multiagent communication with backpropagation. In

A Large-Scale Computational Model of Conceptual Blending Using Multiple Objective Optimisation

João Gonçalves*, Pedro Martins¹, and Amílcar Cardoso¹
¹University of Coimbra, CISUC, DEI
{jgone@dei.uc.pt, pjmm@dei.uc.pt, amilcar@dei.uc.pt}

Abstract

We present a Computational Creativity (CC) system ¹ based on Conceptual Blending (CB) theory. To obtain diverse output – in the form of blend spaces – it allows a trade-off in the optimisation process according to multiple criteria. It is handled by a high performance Multi Objective Evolutionary Algorithm (MOEA) supporting a large Knowledge Base (KB), large input spaces, a high number of mappings as well of frames, all in the range of millions of elements in their structures. Some of CB’s Optimality Principles (OPs) were adapted into objectives as well as new ideas we decided to investigate. Initial experiments allows us to conclude that the system exhibits a form of creativity and its output is capable of depict a simple drawing or a short story.

Introduction

CB theory (Fauconnier and Turner 2002) was proposed as a cognitive theory that explains mechanisms involved in the creation of meaning and insight in the human mind. Those mechanisms are likely to be involved in creation of meaning, argumentation and the communication of thought (Coulson 2006). In the last decade CB has been successfully used in various international computational systems such as (Schollemmer et al. 2014) and (Martins et al. 2019).

One of the first computational systems to be based on CB was Divago (Pereira 2005) and it is, to the best of our knowledge, the only system to date that thoroughly formalises the OPs and studies their impact on the generated blend spaces.

Divago was followed by a CC system recreating from scratch (Gonçalves, Martins, and Cardoso 2017), with the purpose of going beyond simple toy problems. It achieves that goal by scaling up the quantity of manipulated data: Input Spaces, mappings and frames. In that system the various qualities of the blend space (such as the OPs and other aspects) are combined using predefined weights and evaluated as a single fitness function. However, combining multiple objectives in a single weighted function requires a careful selection of those weights which if incorrectly done can present the user with disparate search spaces (Konak, Coit, and Smith 2006). Additionally, the more objectives there are the more time consuming that selection is.

To solve the above issue our latest computational CB system makes use of Multi Objective Optimisation (MOO). The output of the new system is scattered throughout a Pareto front which contains a diverse set of solutions according to the multiple objectives. Some of these are CB’s Optimality Principles, others are qualities we expect the blend spaces must have. Like its antecessor, the system supports a large KB as the source of Input Spaces (ISs), a high number of mappings and of frames. This allows for a even wider diversity in the system’s output.

Being a Computational Creativity system, we expect our implementation to exhibit some form of novelty and practicality (Ritchie 2001). The former means that it should create new statements not existing in its KB. The latter that its output should be easy to interpret and be useful, e.g., to serve as a sketch for a drawing or a short story.

The document is organised as follows: after this introductory section we follow with a short description of CB, then with the details of our CB framework and the implemented objectives. It is followed by the section where we justify the usage of a MOO algorithm. Later, we present some of the results, followed by the conclusions of this work and what we expect to do in the future.

Conceptual Blending Theory

The theory involves interactions between four mental spaces: two input spaces, a generic space and the blend space (Fig. 1), the latter of which contains the product of the CB process. A mental space corresponds to a partial and temporary structure of knowledge built for the purpose of local understanding and action. The ISs supply the content that will be blended. Then, a partial mapping is formed between the ISs. This mapping serves as a sense of similarity (an analogy) between the two spaces and is reflected in the generic space containing the elements shared by the ISs. A selection of this mapping is used to partially project structures from both ISs – including nearby elements – integrating them in an emerging structure called the blend space.

The blend space integrates elements in three sub-tasks: the projection of elements from the ISs into the blend space (Composition); using existing knowledge in the form of background frames to generate meaningful structures in the blend (Completion); executing cognitive work in the blend according to its ongoing emergent structures (Elaboration).
CB theory mentions *frames* to guide the blending process towards stable and recognisable wholes. Frames represent situations, stereotypes, interactions or recurring patterns of some sort involving various participants.

The CB process is guided towards highly integrated and coherent blends through eight OPs: **Integration**, **Topology**, **Web**, **Relevance**, **Unpacking**, **Intensification of Vital Relations**, **Maximisation of Vital Relations** and **Pattern Completion**. The reader is redirected to (Fauconnier and Turner 2002; Geeraerts 2006) to know more about the OPs.

**Implementing Conceptual Blending**

To create the two ISs we use a KB of facts given as a set of semantic relations between pairs of concepts. The two ISs are identified using the concept pairs present in the mapping (explained below). The KB and the mental spaces are represented as a semantic graphs of the form $G(V,E)$ with $V$ the set of vertices (concepts) and $E$ the set of ordered (and labelled) pairs of vertices. The $n$ edges are stored as triples of the form $(\text{concept}_{i,1}, \text{relation}_i, \text{concept}_{i,2})$, e.g. $\text{wing}, \text{partOf}, \text{bird}$.

Multiple mappings are supported. An individual mapping associates two different subsets of concepts from each IS. A mapping $M$ is defined as a set of $n$ ordered concept pairs, $M = \{p_1, \ldots, p_n\}$, $p_i = (\text{concept}_{i,1}, \text{concept}_{i,2})$. Each pair is required to have an order in its components to identify to which IS (1 or 2) does concept component belong to. The first concept always exists in the first IS and the second concept in the other IS. An IS can be identified as the subset of edges of the KB containing either all the first or second concepts of the mapping.

Frames are handled as semantic graphs having variables as its vertices. They are converted to Datalog queries with the purpose of identifying the pattern represented by the frame in the blend space. Each edge of the frame is converted to a Datalog term, the relation to a predicate and each vertex to a *unique* variable. Then all the terms are merged into a conjunctive query. As an example (Fig. 1), the three edges drawn in cyan in the blend space represent a frame converted to the Datalog query:

$$
\text{purpose}(W,F), \text{ability}(H,F), \text{partOf}(W,H), W! = F, W! = H, H! = F.
$$

Before querying the Datalog engine the blend space is converted to a set of facts. Then, if the query is satisfied by the facts we conclude that the frame exists in the blend space.

In each epoch the Evolutionary Algorithm (EA) evolves a set of solutions using a mutation operator. Each solution contains in its “chromosome” a blend space and one copy of a mapping chosen randomly from the initially given set of mappings. The mappings are not changed during EA’s execution and each blend space evolves according to the same assigned static mapping.

Solutions are either created when the EA initialises the population or some of the existing solutions are unable to be improved. Then, either the mutation picks up a newly created solution or an existing one from the population. The mutation randomly decides to add an edge to the blend space or to remove an existing edge. The removal is straightforward, giving priority to concepts with a high degree to minimise the fragmentation of the blend space into multiple graph components. The addition follows CB’s “guidelines” by selectively projecting one edge in three possibilities: an edge is simply projected from the KB; an edge is projected from one of the ISs or; an edge is projected from both ISs connecting two concepts of a concept pair (defined in the chromosomes’ mapping). In all three cases the EA chooses the new edge with a high probability of connecting to an existing edge of the blend space. When the projection of the edge uses the mapping one of its concepts is chosen to be: the first concept of the concept pair; the second OR; a blend of both concepts in the form “$\text{concept}_{i,1}[\text{concept}_{i,2}]$”.

The mutation is followed by the evaluation of the objectives. The solutions are split in $n$ blocks, given to different executing $n$ threads and evaluated in parallel. At the same time, $n$ Datalog engines try to satisfy the frames (converted to queries and previously cached) using the recently generated blend spaces (converted to databases of facts). To evaluate the novelty of the blend space we count the percentage of new edges of the blend not existing in the KB nor in the ISs. An edge is new if it was projected to the blend space using a mapping (i.e., exchanging concepts from a mapping pair or blending them).

**Objectives to Optimise**

In this implementation the blend space is not required to satisfy all of CB’s OPs. We implement those we deem appropriate for our intent and create additional objectives to make the blend space have aspects which do not seem to be described in CB theory. The objectives to be optimised are evaluated as real numbers. Depending on the objective, it is either maximised or minimised (the latter is equivalent to the maximisation of its additive inverse). What follows are the implemented objectives and their purpose:
(1) **Number of Integrated Frames** - follows CB’s Integration principle. It reflects the number of frames present in the blend space. A frame in the form of a Datalog query is integrated in the blend if the query is satisfied by the blend space after this being converted to a set of statements (see Fig. 2). In our opinion, a blend space with many integrated frames is undesired because it can represent almost any conceptual structure such as a situation, an event, a stereotype, etc. Hence this objective is to be minimised while requiring it to be at least one.

(2) **Number of Constraints of the Largest Frame** - Fauconnier and Turner’s Pattern Completion principle influences this objective. We interpret a frame as the composition of multiple conditions (semantic relations) such as X is a Y and/or X moves along path Z. Given a high amount of frames (thousands or more) spanning a diverse number of conditions and a restricted number of semantic relations, a larger frame (according to its number of conditions) will very likely contain the same conditions as a smaller frame (the example two conditions given above are very likely to exist in many frames). By having a higher number of conditions a frame restricts the blend space to a more specific situation or mental structure, something we find more interesting. Therefore from all the frames integrated in the blend, this objective aims to maximise the number of conditions of the largest frame.

(3) **Mean of Within-Blend Relation Semantic Similarity** tries to bring some variety to our CB’s experiments. We examined a simple application of semantic similarity to the relations present in the blend space by “complementing” the relations of connected pairs of edges. For example, this objective would prioritise a blend space with the relations partOf and ableTo connected instead of partOf linked to another partOf (or a memberOf). Our reasoning is based on the principle that in this example the former two relations should be more semantically different than the latter two. Given a word embedding containing the relations present in the ISs the algorithm creates a list of semantic cosine similarities (range between -1 and 1) using the two connected relations of all the pairs of connected edges in the blend space; finally it calculates the arithmetic mean of the list. In order to minimise the similarity this objective is also to be minimised.

(4) **Mean Importance of Vital Relations** - CB mentions certain types of relations - Vital Relations - such as identity, cause-effect, part-whole, space, property, etc. The presence of those relations in the blend is desirable and they allow for the compression of information present in the ISs. CB describes two OPs related to vital relations - the Maximisation and the Intensification of Vital Relations. Our implementation does not support compression (or intensification) but it does allow for the frequency of some relations to be maximised in the blend space. To achieve this a previously created table mapping an edge label to a weight is used with a higher value meaning the given relation should be more frequent. Then, a weighted sum is calculated using the weight of the relation in the table and its relative frequency in the edges of the blend space. The latter sum is then to be maximised.

(5) **Input Spaces Balance** prevents the blend space of being the projection of only one (or none) of the ISs. When this objective is maximised the blend contains the same number of concepts projected from the first IS as from the second IS. With $c_k$ the number of concepts of the $k \in [1,2]$ IS the objective is $\min(c_1,c_2)$ divided by $\max(c_1,c_2)$. This objective has two remarks: when a concept is a blend of two concepts from the first and second ISs, e.g., “surgeon | butcher”, it increases both $c_1$ and $c_2$ concept counters; if either $c_1$ or $c_2$ are zero the objective is zero to prevent division by zero.

(6) **Input Spaces Intertwined Mix** - it interconnects concepts of different ISs throughout the emerging blend space, i.e., any concept of one IS should be connected to a concept of the other IS. It is calculated by counting all the edges of the blend space which connect two concepts of different ISs and dividing the count by the number of edges of the blend space. This value is to be maximised.

(7) **Mean of Words per Concept** - in previous experiments we noticed that the blend space tended to have large concepts such as “tybee island strand cottage historic district”, “south african class exp 2 2-8-0” or a blend composition of both. Intuitively large concepts are likely to be distinct (the latter example corresponds to a specific railway locomotive). However, larger concepts complicate the interpretation of the blend and therefore an objective was created to help prevent these situations. Hence the addition of an objective to minimise the arithmetic mean of the number of words of each concept in the blend space. The words are counted by splitting the concepts in tokens separated by a white-space character or equivalent (underscore, question mark, etc.).

(8) **Blended Concepts Ratio** - also noted in previous experiments was that in order to optimise for either the Input Spaces Balance or Input Spaces Intertwined Mix objectives, the blend space tended to be composed of mostly (or only) blended concepts, e.g. “surgeon | butcher” and “scalpel | knife” - an interesting side effect of using an EA. We added a objective to fix this issue by minimising the ratio of blended concepts to the total number of concepts of the blend space. However we agree that it is not a perfect solution as it reduces (and sometimes completely removes) the presence of blended concepts.
CB as Multi Objective Optimisation

The blend spaces generated by our system must satisfy at least one of objectives. The optimal solutions found by the MOO framework are Non Dominated (or Pareto Optimal) if none of the objectives can be improved without worsening some of the other. As the MOO framework a MOEA is used having the advantages of relaxing the search for the overall optimum while reducing the time to obtain results we deem good enough. This is of especial importance in our case as we are using a large KB as the source of knowledge. The EA also easily allows the search domain (the blend space) to be a complex structure such as a semantic graph.

Previous work and Early Experimental Results

The KB supplying the facts was a custom version of ConceptNet V5 (Speer and Havasi 2012) with 1 229 508 concepts, 1 791 604 relations and 39 types of relations 2.

A set of 1404 mappings was extracted from the KB using (Gonçalves, Martins, and Cardoso 2018) 3. This mapping framework runs a Genetic Algorithm (GA) to find mappings between the ISs using relation based isomorphisms. On average, each mapping had 1413 ± 1188σ concept pairs.

We used (Gonçalves et al. 2019) 4 to generate the frames. Based on a MOEA, it scholastically generates patterns from existing structures in the KB and checks for the prevalence of those patterns. The idea is that recurrent patterns should represent frames. Resulting patterns are evaluated and selected according to different criteria. Using this framework 76187 frames were extracted from the former KB having on average 6.3±1.5σ edges, 6.7±1.4σ variables and a Relation Semantic Similarity of 0.34 ± 0.1σ.

epsilon-NSGA-II was used as the EA. The population size was constant at 2048. Multiple runs were executed where each took the required epochs to reach on average a total of 12±6 hours. The blend spaces were evolved during 20 000±15 000 epochs. The solution’s blend spaces were required to have at least 2 integrated frames, an input space balance of at least 10% and the number of concepts between 3 and 10, with this last upper limit set for an easier interpretation of the generated blend space. Otherwise the number of concepts can grow to a large quantity.

A summary of the results is shown in Table 1 and some examples in Fig. 3. In most of the blend spaces relations between typically unrelated concepts can be seen, e.g. kissing someone is like a mountain, an amphibian made of cold blooded-steel and a tree made of perforated paper capable of cooling food, demonstrating objectives 5 and 6 – a intermix of both input spaces, the usage of mappings and the creation of analogies and of simple metaphors. Most blend spaces generated by the system have vital relations (required by the fourth objective): made of and part of (part-whole), capable of (cause-effect) and isa (identity) are some of the examples. Objectives 7 and 8 clearly minimised the number of words in the concepts and the presence of blended concepts in the blend space, respectively. The Mean of Within Blend Relation Semantic Similarity objective brought variety to the presence of somewhat dissimilar connected relations in the blend spaces. It is more likely they contain relations with different labels instead of being dominated by a single relation of the highest importance (because of the fourth objective).

Further Work

During the implementation and experimentation of this work we noticed several questions. These are the following:

- we think that the way novelty is measured is incomplete as it does not take into account a partial change of the information projected from the ISs. Hence a new way of calculating novelty should be investigated,
- improve the mapping’s quality by incorporating semantic or ontological knowledge between the involved concepts and nearby components. This would allow the system to relate domains of knowledge further disconnected,
- compression of Vital Relations could be implemented through different layers of semantic processing,
- somewhat related to previous question, some form of concept substitution / compression could be applied to concepts related by a relation of semantic similarity or of difference, e.g., replacing a concept by its synonym or antonym. This will likely create further novelty and new relations of interest,
- investigate the EA’s optimisation shortcut addressed by the objective “Blended Concepts Ratio” and a better way of preventing the blend space of being dominated by blended concepts,

Figure 3: An example of four of the generated blend spaces.
Table 1: Statistical properties of the optimised objectives, novelty and other blend space’s properties. It contains 7386 solutions.

<table>
<thead>
<tr>
<th>Objective / Property</th>
<th>Minimum</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Integrated Frames</td>
<td>0</td>
<td>12.369</td>
<td>15.417</td>
<td>132</td>
</tr>
<tr>
<td>Number of Constraints of the Largest Frame</td>
<td>2</td>
<td>4.4356</td>
<td>1.2252</td>
<td>8</td>
</tr>
<tr>
<td>Mean of Within-Blend Relation Semantic Similarity</td>
<td>0.03095</td>
<td>0.23884</td>
<td>0.16251</td>
<td>0.89158</td>
</tr>
<tr>
<td>Mean Importance of Vital Relations</td>
<td>0.07</td>
<td>0.74376</td>
<td>0.19007</td>
<td>1</td>
</tr>
<tr>
<td>Input Spaces Balance</td>
<td>0.14286</td>
<td>0.86858</td>
<td>0.14016</td>
<td>1</td>
</tr>
<tr>
<td>Input Spaces Intertwined Mix</td>
<td>0</td>
<td>0.71975</td>
<td>0.22067</td>
<td>1</td>
</tr>
<tr>
<td>Mean of Words per Concept</td>
<td>1</td>
<td>1.7680</td>
<td>0.6487</td>
<td>4.875</td>
</tr>
<tr>
<td>Blended Concepts Ratio</td>
<td>0</td>
<td>0.28151</td>
<td>0.18553</td>
<td>1</td>
</tr>
<tr>
<td>Number of concepts in the blend space</td>
<td>3</td>
<td>5.8738</td>
<td>2.3779</td>
<td>10</td>
</tr>
<tr>
<td>Number of edges in the blend space</td>
<td>2</td>
<td>6.5757</td>
<td>4.3176</td>
<td>17</td>
</tr>
<tr>
<td>Novelty</td>
<td>0</td>
<td>0.79773</td>
<td>0.20259</td>
<td>1</td>
</tr>
</tbody>
</table>

- instead of requiring strictly equal semantic relations allow the matching of various levels of similar semantic relations in the mapping, frame mining and CB implementation (e.g. matching “partOf” to “component of” or to “belongs to”).

Lastly, a study should be done about how the presence of frames in the blend space influences its quality in some aspect(s) - if positively, negatively or indifferently. Are frames really required or could both the input spaces and mappings contain all knowledge that is required for a good blend?

Conclusions

We have presented a computational implementation of a MOO system which uses CB theory. The blend’s space criteria in the form of objectives to be optimised were also explained. The system was demonstrated to be capable of handling a large amount of knowledge in the form of input spaces, mappings and frames while at the same time exhibiting creativity in the generated blend spaces.

Acknowledgements

This work is funded by national funds through the FCT - Foundation for Science and Technology, I.P., within the scope of the project CISUC - UID/CEC/00326/2020 and by European Social Fund, through the Regional Operational Program Centro 2020.

João Gonçalves is funded by Fundação para a Ciência e Tecnologia (FCT), Portugal, under the PhD grant SFRH/BD/133107/2017.

References


Wölflin’s Affective Generative Analysis for Visual Art

Divyansh Jha Hanna H Chang Mohamed Elhoseiny
King Abdullah University of Science and Technology (KAUST)
{divyansh.jha, hanna.chang, mohamed.elhoseiny}@kaust.edu.sa,

Abstract
We propose Wölflin Affective Generative Analysis (WAGA) as an approach to understand and analyze the progress of machine-generated artworks in contrast to real art and their connection to our human artistic heritage, and how they extend the shape of art history. Specifically, we studied the machine-generated art after integrating creativity losses in the state-of-the-art generative models e.g., StyleGAN v1 and v2. We denote these models as Style Creative Adversarial Networks v1 and v2; in short, StyleCAN v1 and v2. We contrasted the learned representation between real and generated artworks through correlation analysis between constructed emotion (collected through Amazon MTurk) and Heinrich Wölflin (1846-1945)’s principles of art history. Analogous to the recent ArtEmis dataset, we collected constructed emotions and explanations on generated art instead of real art to study the contrast. To enable Wölflin Affective Generative Analysis, we collected 45,000 annotations (1800 paintings × 5 principles × 5 participants) for each of the five Wölflin principles on 1800 artworks; 1000 real and 800 generated. Our analysis shows a correlation exists between the Wölflin principles and the emotions. The collected dataset, analysis, and code is made publicly available at https://vision-cair.github.io/WAGA.

Introduction
With the development of computational creativity, machines are capable of classifying real artworks styles. However, the machine’s ability to assess and classify its AI-generated artworks is less understood and requires further scholarly scrutiny (Colton 2008). The main question we address in this paper is to quantitatively and qualitatively analyze the contrast between real and generated artworks from deep neural representations’ perspectives. We study AI-generated art in three analysis dimensions: 1) Likeability evaluated by human ratings, 2) Their learning representation connection to Wölflin’s principles (Wölflin 1915), 3) emotions constructed by human participants. In contrast to (Elgammal et al. 2018), which used only real art for its analysis, our study focuses on the contrast between real and AI art generated using state-of-the-art GAN models, i.e., StyleGAN1 (Karras, Laine, and Aila 2019), and StyleGAN2 (Karras et al. 2020). We add the CAN loss (Elgammal et al. ;

Figure 1: AI art constructing diverse emotional experiences.

Sbai et al. 2018) to these architectures. We denote the corresponding models as StyleCAN1 and StyleCAN2. We also collect data of Wölflin’s principles on 1000 real art pieces and 800 generated art pieces from StyleGAN2 and StyleCAN2. Results of our study provide insights into the emotion of generated artworks. Figure 1 shows examples of AI-generated artwork with StyleCAN2 that constructs emotional experiences in survey participants.

Contribution: (1) We introduce StyleCAN v1 and v2 by integrating the CAN loss StyleGAN v1 and StyleGAN v2 models and observe that StyleCAN v1 and v2 have higher mean average likeability compared to StyleGAN v1 and StyleGAN v2. (2) We present a novel study on how AI Art generative models learn inherent features of our art heritage like Wölflin’s principles. We also study the ability of these models to constructs our emotional experiences compared to real art. (3) We collect Wölflin principles annotations on real and AI art. We also collect emotion labels and their explanation on AI art. (4) Using the collected data, we performed detailed analysis that contrast real art and AI art based on Wölflin’s principles, constructed emotion categories, and corresponding explanations. We also observed connections between Wölflin’s principles and the constructed emotional experiences. Since we study both the Wölflin’s and the affective perspective for visual art, we denote our approach to analyse visual art
Is this painting Linearly or Painterly?
Do not submit the response without understanding the difference between the two art principles.
If you are not a proficient English speaker, please don’t accept this HIT.
Thanks a lot for your hard work!

Figure 2: The data collection interface of Wölfflin’s principles showing the examples used to train surveyors to classify between the two opposing concepts: linearly and painterly.

as Wölfflin’s Affective Generative Analysis (WAGA) and hope this may encourage future comprehensive analysis of machine-generated art.

**Related Work**

Existing literature on AI for the art creation process has shifted from being emulative to being more creative (Elgammal et al.; Hertzmann 2020; Sbai et al. 2018). Although recent creative AI models can produce novel quality artworks, it is less understood whether these models have all the characteristics of a creative system. (Colton 2008) defined three main characteristics that creative systems should have: (a) “the ability to produce novel artifacts (imagination), (b) “the ability to generate quality artifacts”, and (c) “the ability to assess its creation”. Wölfflin’s Principles of Art History (Wölfflin 1915) are one of the key methodologies in art history that differentiates art styles. They have five categories to classify the stylistic component of the painting: 1) Linear and Painterly, 2) Planar and Recessional, 3) Closed Form and Open Form, 4) Multiplicity and Unity, and 5) Absolute Clarity and Relative Clarity. (Elgammal et al. 2018) has demonstrated a connection between Wölfflin’s Principles and machine’s learning representations of artworks. Specifically, Wölfflin’s Principles were shown to be implicitly learning that each principle was shown to have a strong correlation by one or more neurons in their Neural Network. Our work extends this analysis by collecting performing Wölfflin’s Principles analysis on Machine generated artworks.

The theory of Constructed Emotions (Barrett 2017) suggested that emotions are constructed rather than triggered. In line with the theory, (Achlioptas et al. 2021) collected responses of emotions constructed by Human participants who get exposed to real artworks from the WikiArt dataset. Our work also aims at understanding how people construct emotions from visual art created by AI and contrast that to real art. Hence, we collected emotional responses for generated art as well.

**Data Collection**

Heinrich Wölfflin proposed five principles for visual art (Wölfflin 1950):

1. **Linearly and Painterly**: Linearly paintings depict isolated objects and clear boundaries and have all the figures illuminated. Painterly depicts blurry outlines and swift brushstrokes.
2. **Planar and Recessional**: The composition of objects in planar are arranged in planes parallel to the plane of the canvas. In recessional, these objects can be in angle and focus on spatial depth.
3. **Closed-form and Open-form**: All figures in closed-form are balanced within the frame, while in open-form, the figures are cut off. While the former is mostly self-contained, the latter indicates space beyond the frame.
4. **Multiplicity and Unity**: In multiplicity, we have distinguished parts, and each part demonstrates independent
features. In unity, figures weld together, and colors blend in.

5. Absolute clarity and Relative clarity: While absolute clarity has realistic representation, relative clarity has representations enhanced with visual effects.

We collected Wolfflin’s principles annotations by training people to learn one of these principles and then ask them to identify it in a painting. For example, the survey interface design provides descriptions of Wolfflin’s principles (one at a time). Figure 2 shows the interface design for identifying linearly and painterly paintings. Based on the shown explanation of linearly and painterly paintings, the viewer selects a rating scale from 1 to 5 (1: Clear Linearly, 2: Mostly Linearly, 3: Borderline, 4: Mostly Painterly, 5: Clear Painterly). Using this method, we train the survey participants. We average the five ratings for every artwork and normalize the resulting score between 0 and 1, where normalized scores closer to 0 are linear, and scores closer to 1 define painterly characteristics. This way, an artwork has five floating-point characteristics. This way, an artwork has five floating-point characteristics. This way, an artwork has five floating-point characteristics. This way, an artwork has five floating-point characteristics. This way, an artwork has five floating-point characteristics. This way, an artwork has five floating-point characteristics.

We conduct these experiments for both real and generated arts for each Wolfflin principle. We release the web interfaces for collection of the Wolfflin principles here.

### Methodology

Generative Adversarial Networks (GANs) (Goodfellow et al. 2014; Radford, Metz, and Chintala 2015; Ha and Eck 2018) is a popular modeling choice. However, the classic GAN training objective does not promote the generation of novel content beyond the training data. A GAN trained on artwork can generate Da Vinci’s “Mona Lisa” again, but it will not produce a painting of a new style. Recent work has been able to encourage GANs to produce novel images. Inspired by (Elgammal et al.; Sbai et al. 2018), we adapted GANs to generate novel paintings by encouraging the model to deviate from existing art styles. We attach a head on the GAN’s Discriminator $D$, which predicts the style of an art piece. The Generator is then encouraged to generate real-looking examples, which is hard for $D$ to assign a class.

**StyleGAN Model (Figure. 3):** We train a StyleGAN model (Karras, Laine, and Aila 2019), (Karras et al. 2020) using the creativity loss (Sbai et al. 2018) on the WikiArt dataset. In contrast, (Sbai et al. 2018) experiments it for fashion dataset. Concretely, our generator loss becomes:

$$L_G = L_{StyleGAN} + \lambda L_c$$

$$L_D = L_{D_{StyleGAN}} + \lambda L_s$$

where $L_G$ and $L_D$ are the losses for the generator and discriminator, respectively, $\lambda$ is a hyperparameter, $L_c$ is the creativity loss, and $L_s$ is the style classification loss.

In Table 3, we find that participants prefer generations from models integrated with CAN loss more than the generations from vanilla versions. The mean likeability of StyleGAN1 is 2.5% more than its vanilla counterpart. For StyleGAN2, it is 7.0%. We also find that more people think that an artist created the art by an artist or machine (we name this Turing Test).

**Likeability Experiment:** We follow (Elgammal et al.) for likeability experiment. We ask participants two questions. Q1) Rate the art on the scale of 5. Q2) Whether art is created by an artist or machine (we name this Turing Test).

**Emotion Experiment:** Follow (Achlioptas et al. 2021) setup, we ask participants to select one of 9 emotions (amusement, awe, contentment, excitement, anger, disgust, fear, sadness, and something else). We also ask them to describe in text what made them feel so.

### Results

In Table 3, we find that participants prefer generations from models integrated with CAN loss more than the generations from vanilla versions. The mean likeability of StyleGAN1 is 2.5% more than its vanilla counterpart. For StyleGAN2, it is 7.0%. We also find that more people think that an artist creates generations from the model trained with CAN loss for both StyleGAN1 and StyleGAN2 models. The generated images achieved a higher likeability score than Art Basel.

Classifier features’ principal components correlation with Wolfflin’s principles: We find that learning to classify style makes the model also learn Wolfflin’s principles inherently. We compute the Pearson’s correlation coefficients of features of various architectures on real art and generated art computed for all the Wolfflin principles.

### Table 1: Pearson’s correlation coefficients of features of various architectures on real art and generated art computed for all the Wolfflin principles

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Real Art</th>
<th>StyleGAN2</th>
<th>StyleGAN1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet101 + 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ResNet50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VGG16</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Methodology**

Generative Adversarial Networks (GANs) (Goodfellow et al. 2014; Radford, Metz, and Chintala 2015; Ha and Eck 2018) is a popular modeling choice. However, the classic GAN training objective does not promote the generation of novel content beyond the training data. A GAN trained on artwork can generate Da Vinci’s “Mona Lisa” again, but it will not produce a painting of a new style. Recent work has been able to encourage GANs to produce novel images. Inspired by (Elgammal et al.; Sbai et al. 2018), we adapted GANs to generate novel paintings by encouraging the model to deviate from existing art styles. We attach a head on the GAN’s Discriminator $D$, which predicts the style of an art piece. The Generator is then encouraged to generate real-looking examples, which is hard for $D$ to assign a class.

**StyleGAN Model (Figure. 3):** We train a StyleGAN model (Karras, Laine, and Aila 2019), (Karras et al. 2020) using the creativity loss (Sbai et al. 2018) on the WikiArt dataset. In contrast, (Sbai et al. 2018) experiments it for fashion dataset. Concretely, our generator loss becomes:

$$L_G = L_{StyleGAN} + \lambda L_c$$

$$L_D = L_{D_{StyleGAN}} + \lambda L_s$$

where $L_G$ and $L_D$ are the losses for the generator and discriminator, respectively, $\lambda$ is a hyperparameter, $L_c$ is the creativity loss, and $L_s$ is the style classification loss.

In Table 3, we find that participants prefer generations from models integrated with CAN loss more than the generations from vanilla versions. The mean likeability of StyleGAN1 is 2.5% more than its vanilla counterpart. For StyleGAN2, it is 7.0%. We also find that more people think that an artist creates generations from the model trained with CAN loss for both StyleGAN1 and StyleGAN2 models. The generated images achieved a higher likeability score than Art Basel.

Classifier features’ principal components correlation with Wolfflin’s principles: We find that learning to classify style makes the model also learn Wolfflin’s principles inherently. We compute the Pearson’s correlation coefficients of features of various architectures on real art and generated art computed for all the Wolfflin principles.
Table 2: Weights of the linear classifier when trained on Wöllfin’s principle to different emotions. The term “vs” is used in the table to compare opposing concepts of each Wöllfin’s principle.

<table>
<thead>
<tr>
<th>Emotion</th>
<th>Linearly vs Painterly</th>
<th>Painterly vs Recessional</th>
<th>Closed Form vs Open Form</th>
<th>Multiplicity vs Unity</th>
<th>Absolute Clarity vs Relative Clarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amusement</td>
<td>-0.524</td>
<td>-0.2292</td>
<td>0.173</td>
<td>-0.229</td>
<td>-0.333</td>
</tr>
<tr>
<td>Fear</td>
<td>-0.209</td>
<td>-0.2292</td>
<td>-0.921</td>
<td>-0.42</td>
<td>-0.06</td>
</tr>
<tr>
<td>Disgust</td>
<td>0.089</td>
<td>-0.2292</td>
<td>0.301</td>
<td>-0.301</td>
<td>0.994</td>
</tr>
<tr>
<td>Anger</td>
<td>0.266</td>
<td>0.096</td>
<td>0.807</td>
<td>0.807</td>
<td>0.369</td>
</tr>
<tr>
<td>Sadness</td>
<td>0.266</td>
<td>0.285</td>
<td>-0.386</td>
<td>-0.386</td>
<td>0.07</td>
</tr>
<tr>
<td>Mixed Feelings</td>
<td>-0.001</td>
<td>-0.2292</td>
<td>-0.2292</td>
<td>-0.2292</td>
<td>0.001</td>
</tr>
<tr>
<td>Something else</td>
<td>-0.001</td>
<td>-0.2292</td>
<td>-0.2292</td>
<td>-0.2292</td>
<td>0.001</td>
</tr>
</tbody>
</table>

Figure 4: Distribution of emotions by emotion survey experiments on both real and generated art.

similar percentage in StyleGAN2 (7.9%). Fear emotion percentage increased from 9.1% for real art to 15.8% for StyleGAN2 and 12.9% for StyleCAN2.

Likeability experiment qualitative analysis: We observe that AI-generated paintings with high likeability (Q1) score and high Turing test percentage (Q2) in table 3 were from NN↑ group. This shows that the generated is both novel (because of high NN distance) and likeable. We can see some examples in Figure 5. Artworks with high likeability in the NN ↓ group were mostly natural landscapes, vibrant colors, and distinct brushstrokes. They looked like Monet’s “Impression Sunrise.” NN ↑ group portrayed abstract figures with contrasting solid colors. Artworks from the NN ↓ group, which had a low Turing test percentage were abstract paintings, reminding the viewer of Mondrian’s “Broadway Boogie Woogie.” The style of these paintings may have prompted survey participants to assume a machine did these paintings, whereas artworks from NN ↑ with lower Turing test numbers looked like images seen under a microscope. The paintings’ cell-like, scientific feel made the paintings appear like screenshots of microscopic scans, contributing to a lack of artistic intent.

Emotions experiments qualitative analysis: We derived and analyzed common elements of the AI-generated artworks that constructed various emotions.

Awe: Artworks that portrayed familiar subject matters like “brown coat”, “man”, “plant”, and “sky” tend to construct emotions of awe within the audience. Many participants referred to the use of complementary colors and soft color scheme to be the underlying cause that constructed emotions of awe; example response by our participants: “the play of colors in the sky of this painting is magnificent”).

Anger: The disorderly arrangement of figures tends to create a sense of unease and construct anger (e.g., “The painting looks confusing and shows no representation”). The confusion caused by the abstract subject matter caused discomfort and points out the participants’ annoyance in understanding the meaning of the painting (e.g., “The color (red) used seems like a man bleeding with anger”).

Contentment: We observed that floral color schemes constructed contentment emotions, as quoted by the participants (e.g., “mix of colors between green and yellow..."
minds the changing of seasons, brings peace and tranquility”). Participants also underlined the role of depth, layers, and orderly composition played in constructing their awe feeling (e.g., "This painting makes me feel relaxed because the items are well ordered and displayed in a coherent fashion"). Positive past experiences evoked by the artworks contributed to the participant’s selection of awe (e.g., "I liked the image I felt pleasure because remember my childhood").

**Amusement:** Artworks associated with feelings of amusement interestingly reminded the audience of animals (e.g., “The white cat is hiding behind the building”). Participants were also amused by depiction of human characters. The portrayal of human subjects created both a sense of familiarity and beauty which contributed the construction of amusement e.g., “there seems to be a pretty girl dancing across this image”.

**Disgust:** Dark color schemes and visual effects tend to construct emotions of disgust, as stated by the participants: “Too much darkness on the sea”, and "looks like a dark cloud about to eat a human". Also, when an artwork didn’t clearly convey its message or meaning, the audience felt emotions of disgust. Participants stated that the lack of expression came from “a lifeless representation”.

**Discussions**

In our experiments, we notice the following key observations.

1. From Table 1 we observe that the discriminator of GANs learn the Wöllflin principles inherently as we observe stronger correlation values as compared to traditional classifiers.
2. In Figure 4, we find that the generated art works construct diverse set of emotions in the viewer.
3. In Table 2, we also find strong correlation between these emotions and Wöllflin’s principles. We find these principles can be used to predict the emotion of an art piece. Hence, being able to compute specific Wöllflin principle can predict the emotion that the art piece will construct.

**Conclusion**

We introduced Wöllflin Affective Generative Analysis as an approach to understand and analyze machine-generated art-works in contrast to real art. To perform this analysis, we collected Wöllflin principles annotations on 1000 real paintings and 800 generated artworks. We also collected Affective responses on artworks produced by state-of-the-art generative models, including StyleGAN2 and our improved version of it StyleCAN2 after applying CAN loss. We show that models inherently learn the stylistic principles and emotions during the learning process. We showed that Wöllflin’s principle coefficients are similar for generated and real artwork, showing that the generated art also contains the historical styles studied in the past. By training a multi-label classifier to predict emotion from Wöllflin principles, we observed that some Wöllflin principles have solid connections for constructing certain emotions. We release our models, analysis, and data on real and generated art to facilitate future research.

**References**


[Elgammal et al.] Elgammal, A.; Liu, B.; Elhoseiny, M.; and Mazzone, M. Can: Creative adversarial networks, generating “art” by learning about styles and deviating from style norms.


[Ha and Eck 2018] Ha, D., and Eck, D. 2018. A neural representation of sketch drawings. ICLR.


The Humble Creative Machine

Christopher Cassion  
Santa Clara University  
Santa Clara, CA  
ccassion@scu.edu

Margareta Ackerman  
Santa Clara University  
Santa Clara, CA  
mackerman@scu.edu

Anna Jordanous  
University of Kent  
Canterbury, UK  
a.k.jordanous@kent.ac.uk

Abstract

In this paper, we present a fresh perspective at co-creativity - using human development as a blueprint, we argue that fostering human creativity is a natural evolution of creative machines. We introduce the notion of a “humble” creative machine - one that is less concerned with demonstrating its own independence, and instead uses its (potentially advanced) creative abilities to cultivate human creativity. We present characteristics of Humble Creative Machines and offer examples of existing (almost) humble machines.

Introduction

The Association of Computational Creativity website states:

“The goal of computational creativity is to model, simulate or replicate creativity using a computer, to achieve one of several ends:  
... to design programs that can enhance human creativity ” [our emphasis added]

In practice, however, the quoted aim appears to have been de-emphasized over the past decade or two. According to a 2017 review of the field (Loughran and O’Neill 2017), ICCC sessions relevant to this area have shifted away from ‘Creativity Support Tools’ (ICCC’2010) and ‘The Helpful’ (ICCC’2011) towards co-creativity research and research on making creative systems more autonomous. This is also marked by comments in the influential ECAI paper by Colton and Wiggins that redefines computational creativity, in part to emphasise “the difference between the systems we build and creativity support tools studied in the HCI community … and embedded in tools such as Adobe’s Photoshop, to which most observers would probably not attribute creative intent or behaviour. ” (Colton and Wiggins 2012, our emphasis added).

This stage in the development of Computational Creativity mirrors an analogous stage in human development, where a child begins to differentiate from their parents and form their own identity. Erikson’s stages of psychosocial development outline the development of healthy individuals from infancy to adulthood (G.A Oreinstein 2020). Often taking place in adolescence, humans seek to form their own identity (Lewis 2020; Sutton 2021), which necessitates some form of separation from the parents.

Individuation is of course not the final stage of human development. When a person reaches individuation, they soon begin to move towards taking care of others - often in the form of parenting [(Lewis 2020), (Sutton 2021)]. Analogously, we would like to propose that Autonomous Creativity is not the final aim of Computational Creativity.

Another apt analogy comes from academia. A PhD student will often initiate her studies by learning from her advisor, relying on the supervisor’s vision and ideas. As the student progresses in her studies, she gradually develops more of her own ideas, and eventually becomes an independent researcher, often ending up with different views and research interests from their advisor.

If the student stays in academia, before long, she will take on students of her own, and generously share her own vision in order to help the development of her students. Confident in her own research abilities, she lets her students take the spotlight, and likely exhibits less concern when it comes to accurate credit sharing with those she mentors.

In this paper, we propose that Computational Creativity is now sufficiently advanced to take on the mentorship role. This does not mean that autonomous creativity has been fully actualized - much exciting research remains in expanding the autonomy of creative machines, such as elevating their evaluative capabilities. Analogously, Erickson’s stages are inherently fluid and overlapping. A person may encourage the growth of others without abandoning their own development.

The autonomous and mentorship roles need not be conflicting. We propose that the creative engine itself and how it is presented can be viewed as two separate components. The same creative engine can take on autonomous roles as well as supporting roles, perhaps even making the decision on which role it would prefer in different instances.

In this paper, we hope to offer a fresh perspective at co-creativity and encourage the Computational Creativity community to engage in the formation of Humble Creative Machines. We discuss connections to other CC frameworks, and share several systems that already captured some of the vision of Humble Creative Machines.
Relation to Other CC-Models

There has been a great deal of work on interactive creative systems, however the focus is typically on autonomous behaviour within the interaction rather than supportive behaviour (D’Inverno and Luck 2012; Magill and Erden 2012, e.g.), or on the results of the whole collaboration as opposed to results of other individuals within the collaboration (Al-Rifaie, Bishop, and Caines 2012).

Attention has been paid to dialogues and communication between systems (Bown et al. 2020; Saunders et al. 2010) and some work has been done with peers communicating with each other to give feedback (Corneli et al. 2015; Jordanous, Allington, and Dueck 2015), though this paper’s thesis takes this considerably further forward, in terms of using that peer feedback to support the co-creative partner towards greater creativity.

Computational collaborators in human-machine co-creativity frameworks are rarely seen with comparable levels of agency to the human partners (Bown 2015; Jordanous 2017). Resulting limitations and restrictions are being placed on progress in human-machine co-creativity (Kantosalo and Jordanous 2021). However, other fields of creativity research have suggested roles for computational partners that are connected to our suggestions in this paper. For example, take Lubart’s role categorisations of a computational partner in a human-machine co-creativity scenario: the computer as “nanny”, “pen-pal”, “coach” or “colleague” (Lubart 2005). These closely resemble the generally enabling effect of the computational partner on the human collaborator; however Lubart’s roles give no recognition to the importance of the computational partner’s creativity in enabling the human collaborator (Jordanous 2017). The creativity of the computational partner is optional and de-emphasised, whereas in our proposals the creativity of the computational partner is essential; they are a peer that can support others in their creative area.

Casual creators aim to give the user an “intrinsically measurable activity, rather than as an extrinsically-motivated way to accomplish tasks.” (Compton and Mateas 2015). Examples of casual creators place most creative effort on the machines, while enabling an easy and enjoyable way for the user to explore the creative space. Consequently, the user gets to reap the joy, and perhaps even wellness benefits of engaging in a creative activity, but does not necessary grow in their creative abilities. Furthermore, in an interaction where the machine carries most of the creative complexity, dependency on the machine is likely to arise - by contrast, humble machines aim to make the person creatively independent of the machine (see the following section for more details). The goals of casual creators and humble machines are in some sense opposites of each other - casual creators wish to keep the process easy for the user, letting the machines do most of the work, while humble machines help the user gain proficiency until the machine itself becomes unnecessary.

Perhaps the most closely related to our vision is work on Mixed Initiative Co-Creativity. Relying on research from human creativity, (Yannakakis, Liapis, and Alexopoulos 2014) delve in depth into how a creative machine can assist humans in the creative process. In particular, they discuss how the classical iterative process in which the machine takes on a primarily generative role and the user engages in evaluation, is connected with human creative processes such as lateral thinking and creative emotive reasoning.2 Our paper takes on a complementary approach - instead of demonstrating that machines are able to foster human creativity, which was effectively argued by (Yannakakis, Liapis, and Alexopoulos 2014), we focus on how to accomplish this task. In particular, we put forward specific characteristics that machines that aim to foster human creativity should aim for, and present a vision for how powerful creative machines can elevate human creativity.

Humble Creative Machines

In this section, we introduce several criteria for humble creative machines. These criteria, at their core, allow the system and its interactions to focus on the user and the user’s capabilities rather than the machine and what it can independently.

Flexibility

We propose that humble creative machines should be flexible in a couple of ways. The first is flexibility in its range of interaction. Ideally, the system should be able to either do all of the work (autonomous) or none of the work (support tool), and everything in between. The configuration applied in any specific interaction should dependent on the user’s skill level or preference.

For the novice who requires a more guided approach, the system can offer heavy support (ex. only requiring the user to act an evaluate in various stages of the process). Meanwhile, for the expert who only needs occasional inspiration, the system may take a more passive role, and be available for the user as much and when needed.

The second type of flexibility is in the quality of the output. Being capable of sophisticated creative artefacts, the humble creative machine is able to consistently provide expert level engagement to a user. However, the machine should be able to reduce its own level of expertise to better meet the user at their current level of creative development. For instance, a co-creative poetry machine capable of elaborate metaphors may choose to use simpler language that would be a better fit for the users stylistic preference or significantly beyond their current abilities (effective learning takes place when done incrementally).

Learning & Independence

The humble creative machine’s flexibility directly affects its ability to lead its users to learning and fostering independence from the system itself. With flexibility, the system is able to gradually adjust its level of interaction and quality of outputs to meet the user at their level of expertise. This offers a gradual learning apparatus tailored to the user.

---

2We recommend the work of (Yannakakis, Liapis, and Alexopoulos 2014) as complementary reading to the current paper, particularly to those wishing to gain insight into how machines can meaningfully support the human creative process.
to teaching or coaching scenarios, the system can bridge the gap in knowledge and expertise. This can gradually change over time as the user becomes more of an expert and needs the system less and less. As such, it is crucial that the system is able to step back and allow the user to engage more deeply in the creative process as they gain the ability to do so. This may be accomplished through either the system detecting growth in the user, or the user having sufficient control over their interaction with the humble creative machine as to reduce their reliance on it as desired.

Creative
Being a creative system, a humble creative machine should be capable of making creative contributions in its co-creative interactions. At minimum, it should satisfy P-creativity, having the ability to come up with surprising, valuable ideas that are new to itself (Boden 2009). Building on the idea of P-creativity, we further suggest that a humble creative machine should be able to produce output that is surprising, valuable, and new to their human partner.

User Friendly
A user friendly interface with a natural flow, which easily adapts to the user as they grow in their creative abilities, will form the foundation for communicating between the human and the humble machine, allowing for effective learning and growth to take place. At minimum, the co-creative process should not be undermined by unnecessarily complex interactions that would hinder learning and engagement.

Examples of (Almost) Humble Creative Machines
While no previous systems have been created to accurately represent our vision for humble creative machines, some previous systems capture important aspects of this concept. We discuss two machines which approximate our vision.

LyricStudio
LyricStudio is a co-creative lyrics machine. Considering the properties outlines above, the system meets both flexibility criteria. It offers a flexible engagement process, whereby the user chooses how much of its suggestions to utilize, is free to alter any of the recommendations, and may write in their own ideas. LyricStudio also contains options that can make the lyric suggestions be simpler or more complex. The system automatically reflects the user’s writing style, even as it develops over time.

The lyrical suggestions provided by LyricStudio are creative the sense they are novel and useful. In particular, LyricStudio’s generation are novel every time, and useful in the sense that tends of thousands of users have utilized them in the development of their own lyrical material.

LyricStudio has been designed to be highly user friendly consisting of a single minimal page. All of LyricStudio’s capabilities (rhyming, mirroring of language and style, etc) are applied by default, with several advanced features available through a setting panel. The primary interaction with the system is through the “New Suggestions” button, which the user activates as needed.

LyricStudio can be moved closer to the vision of a humble creative machine by increasing its own creative capabilities. Although one can imagine creating an autonomous variation of LyricStudio, this capability has not been developed. As such, the level of assistance that it can provide to novice is limited to providing a single line at a time. It is possible that some new users would benefit from more extensive assistance.

Impro-Visor
Impro-Visor (Kondak et al. 2016; Goldstein et al. 2019) is a music notation tool for producing monophonic lead sheets, specifically intended to help the improviser. Improvisation advice is offered in the form of note coloration, database of licks, and, importantly, automatic lick generation from grammars. We find impro-visor to be a close candidate to being a humble creative machine.

Impro-Visor is an excellent fit for those with some musical expertise. By improvising along with the user, Impro-Visor can give those new to improvisational trading the confidence and experience to grow in their improvisational abilities. Designed for those with some musical experience, Impro-Visor does not currently offer functionality to support those new to music making.

Impro-Visor leads to learning and fosters independence. It offers the experience of learning by doing - users naturally improve through practice. Further, engaging with an creative computational agent eliminates fear of embarrassment, which can make it challenging to master improvisation by practicing with a fellow human musician.

The melody suggestions provided by Impro-Visor are creative artefacts. In trading mode, Impro-Visor offers original melodies that fit with the selected style and the musical session. Using music theory and other domain knowledge, the system is able to provide to the user useful options to finish a desired piece.

The utility of Impro-Visor for supporting human creativity can be greatly improved by making the system more user-friendly. Complex setup and a rich interface can make it challenging to get started and effectively utilize this powerful system. While perhaps less central to CC in general, in the context of enabling human creativity, offering the system through a user-friendly interface can be key for achieving the goals of humble creative machines. There is an opportunity to make Impro-Visor a highly applicable tool for developing musical creativity by improving its setup and simplifying its options, letting the user focus on developing their creative capabilities through a seamless process.

Conclusions and Discussion
In this paper, we introduced a new perspective at co-creativity, as a mature role that can be taken on by creative machine agents. We propose the notion of a “humble” creative machine, which intentionally prioritizes the creative
development of its human partners over demonstrating its own creativity capabilities. Drawing on the works of Erickson’s stages of psychosocial development, we propose that the focus on fostering human creativity through a creative machine marks an advanced stage of development in computational creativity.4

We propose several properties that humble machines should satisfy in their fostering of human creativity. The “humblest” of the machine stems from its willingness to step aside and reduce their own creative contribution when this would better serve to cultivate the creative abilities of their human partner, and fostering independence, that is, helping the user to develop creative abilities to the point that the machines becomes unnecessary.

We presented a couple of examples of systems that capture some key elements of humble machines. However, the broad vision of a humble machines remains open - a machine capable of autonomous creativity that includes the ability, when it so chooses, to apply itself towards the creative development of its human partner, or perhaps even a fellow machine.

References


4It is worth clarifying that co-creativity does not always signal an advanced stage. Human intervention can also utilizes as a necessity to enable a machine to be creative. Using Erickson’s model, we suggest that a return to co-creativity as machine creativity becomes more advanced can represent a meaningful form of progress.


Preface

You hold in your hands the proceedings of the Second Workshop on the Future of Co-Creative Systems. Like the first edition in 2020 the workshop is adjoined with the International Conference on Computational Creativity. This year the workshop is located in Virtual Mexico and held on the 14th of September in 2021.

The emergent discipline of human–computer co-creativity focuses on applied and usable computational creativity systems as well as questions related to collaboration between artificial agents and humans on creative tasks. The workshop aims to bring together researchers from various disciplines to discuss human–computer co-creativity from a range of perspectives. We have offered a venue for presenting on-going work with human–computer co-creative systems, short position papers presenting and addressing important questions and new perspectives for the field, as well as descriptions of projects within research groups.

Our proceedings are peer reviewed and out of the six manuscripts submitted this year, we selected four manuscripts for publication and presentation at the workshop. The papers were first reviewed by the paper chairs, and the papers which fitted the topic of the workshop were then each reviewed by between two or three expert reviewers. Finally, following the recommendations of our expert reviewers our paper chairs performed the final evaluation of the papers and summarized the review results.

We are happy to have attracted papers that all highlight important aspects of human–computer co-creativity research: Three of the accepted papers each address an important principal question in the field of human–computer co-creativity. These include evaluation metrics, models for interaction in human–computer co-creation, and agency in human–computer co-creativity. The final paper examines an important application area for human–computer co-creativity: well being.

Acknowledgements

We thank the Workshop Organizing Committee and the Program Committee for devoting their time and expertise to organizing the meeting this year. We also thank all the authors and workshop participants for being a part of creating a new community for human–computer co-creativity to flourish in.

Workshop Organizing Committee
Anna Kantosalo, Aalto University, General chair
Prashanth Thattai Ravikumar, National University of Singapore, Paper chair
Ollie Bown, University of New South Wales, Paper chair
Kazjon Grace, University of Sydney, Communications chair
Tapio Takala, Aalto University, Organizing committee member
Mary Lou Maher, University of North Carolina at Charlotte, Organizing committee member

Program Committee
Margareta Ackerman, WaveAI, Santa Clara University
Luis Arandas, INESC TEC
Paul Bodily, Idaho State University
Felipe Calegario, Universidade Federal de Pernambuco
João Miguel Cunha, University of Coimbra
Sebastian Deterding, University of York
Jon McCormack, Monash University
Diarmuid O’Donoghue, Maynooth University
Janin Koch, Inria Saclay
Maria Teresa Llano Rodriguez, Monash University
Santiago Negrete-Yankelevich, Metropolitan Autonomous University (Cuajimalpa)
Sarah Vollmer, York University
Evaluating the Effect of Co-Creative Systems on Design Ideation

Jingoog Kim and Mary Lou Maher
Software and Information Systems Department
University of North Carolina at Charlotte
9201 University City Blvd, Charlotte, NC 28223, USA
jkim168@uncc.edu, M.Maher@uncc.edu

Abstract
Evaluating co-creative systems is an open research question in computational co-creativity research. This paper addresses a lack of studies about evaluating the effect of co-creative systems on ideation, a creative process during which designers generate new ideas. This paper describes an approach to measuring ideation as a basis for evaluating co-creative systems in design. Particularly, we are interested in how the contribution of an AI partner in a creative design task influences design ideation in a co-creative system. In order to evaluate a co-creative design system, we present an approach for measuring ideation that has two components: an aggregate analysis and a temporal analysis. With the metrics, we hope to contribute to a critical and constructive discussion on evaluating the impact of AI contributions in other co-creative systems.

Introduction
Computational co-creative systems are a growing research area in computational creativity. While co-creative systems can be applied to a variety of domains associated with creativity and encourage designers’ creative thinking, there are few studies that focus on evaluating co-creative systems. Understanding the effect of co-creative systems in the ideation process can aid in the design of co-creative systems and evaluation of the effectiveness of co-creative systems. However, most research on co-creative systems focuses on evaluating the usability and the interactive experience (Karimi et al., 2018) rather than how the co-creative system influences creativity in the creative process. To evaluate the usability and the user experience of co-creative systems, the studies often used qualitative approaches and a few studies have used a quantitative approach to evaluate the user experience of co-creative systems relying on questionnaires (Kantosalo and Riihiabo, 2019) such as the System Usability Scale (SUS) (Brooke, 1996) and the Creativity Support Index (CSI) (Cherry and Latulipe, 2014). This paper describes a quantitative approach to measure ideation as the basis for evaluating the effect of AI inspiration in a co-creative system. This quantitative approach provides generalized measure for evaluating the effect of co-creative systems in design. With these quantitative measures, researchers can quantify the effectiveness of a co-creative system on design ideation and compare the effect of different AI models in a co-creative system.

Evaluation of Computational Co-Creative Systems
Evaluating co-creative systems is still an open research question and there is no standard metric for measuring computational co-creativity (Karimi et al., 2018). The research on co-creative systems shows various approaches to evaluate co-creative systems and computational co-creativity. Some researchers focus on evaluating the interactive experience and others focus on the effectiveness of the system to produce or generate a creative outcome.

Shimon (Hoffman and Weinberg, 2010) is a robotic marimba player that listens and responds to a musician in real time. The evaluation of Shimon (Hoffman and Weinberg, 2010) is a performance-based evaluation of the system. The evaluation used observation to analyze the system’s behaviors and the audience reactions during the performance. Drawing Apprentice (Davis et al., 2015) is a co-
creative drawing system in which the computational partner analyzes the user's sketch and responds to the user's sketch. Drawing Apprentice (Davis et al., 2015) focused on usability and system accuracy in the evaluation. The evaluation methods include algorithm testing, voting, survey, and retrospective protocol analysis to evaluate the system and interactive experience. Viewpoints AI (VAI) is a co-creative dance partner that analyzes the user’s dance gestures and provides complimentary dance in real-time by a virtual character projected on a large display screen (Jacob et al., 2013a, 2013b). In the evaluation of Viewpoints AI (Jacob et al., 2013a, 2013b), the researchers observed how participants interact with the systems and the participants provided feedback about their interactive experiences. While these examples of evaluation focus on interactive experience in the creative process, Sentient Sketchbook (Yannakakis et al., 2014), and 3Buddy (Lucas and Martinho, 2017), co-creative systems for game level design, focused on the usefulness of the system since both systems support a goal-directed design rather than an open-ended artistic performance. The methods used include a survey, interview, and observation to measure the usefulness of the system. Our study is distinct in evaluating the impact of the AI partner on the user’s ideation process.

Karimi et al. (Karimi et al., 2018) presented a framework for evaluating creativity in computational co-creative systems. This framework responds to four questions that serve to characterize the many and varied approaches to evaluating computational co-creativity: who is evaluating the creativity, what is being evaluated, when does evaluation occur, and how the evaluation is performed. The framework enables comparisons of evaluation focus and methods across existing co-creative systems. Using this framework, we have shown that the evaluations of the existing co-creative systems described in this section respond to “what is being evaluated” with a focus on evaluating the interactive experience and the final product. In this paper, we respond to “what is being evaluated” and “how is the evaluation performed” by evaluating the novelty, variety, and quantity of ideas in the ideation. Section 3 describes how we define and measure ideation in more detail.

**Measuring Design Ideation When Co-creating with an AI Agent**

Ideation is a creative process where designers generate, develop, and communicate new ideas. Ideation in design can lead to innovative design solutions through generating diverse concepts (Akin, 1990; Atman et al., 1999; Brophy, 2001; Cross, 2001; Liu et al., 2003). The goal of design is to develop useful and innovative solutions and design ideation allows designers to explore different areas of the design solution space (Daly et al., 2012; Newell and Simon, 1972). A design process is an evolution of different kinds of representations (Goel and Pirolli, 1992). In a design process, designers externalize and visualize their design intentions and communicate with external visualizations to interact with their internal mental images (Dorta, 2008).

During ideation, designers commonly use freehand sketches and rough physical models as a tool for constructing external representations that also serve as cognitive artifacts of design (Visser, 2006). Making sketches and physical models is an interaction between the designer and the designer’s ideas, similar to a conversation (Dorta, 2008). In the ideation stage, designers frame problems producing new discoveries through this conversation. The graphical and physical representations as cognitive artifacts are essential components of the ideation process.

The first step for measuring ideation is to define what an idea is in the ideation process using a co-creative system. Defining an idea in design ideation using a co-creative system is a challenge since the idea can be defined differently involving the contribution of an AI partner in ideation. In engineering design, an idea is normally considered as a possible solution to a given problem for evaluating the performance of idea generation (Shah et al., 2003). However, an idea can be variously defined as a contribution that contains task-related information, a solution in the form of a verb–object combination, and a specific benefit or difficulty related to the task (Reinig et al., 2007). To define an idea in design ideation using a co-creative system, we use a cognitive approach by collecting think aloud data and coding the data for cognitive issues. We define an idea as a cognitive issue that the designer considers during the design process, and adopt the Function–Behavior–Structure (FBS) ontology (Gero, 1990; Gero and Kannengiesser, 2004) as a basis for segmenting and coding each idea in the design process. FBS ontology is a design ontology that describes designed things, or artifacts, irrespective of the specific discipline of designing. The function (F) of a designed object is defined as its teleology; the behavior (B) of that object is either derived (Bs) or expected (Be) from the structure, where structure (S) represents the components of an object and their compositional relationships. These ontological classes are augmented by requirements (R) that come from outside the designer and description (D) that is the document of any aspect of designing. In this ontological view, the goal of designing is to transform a set of requirements and functions into a set of design descriptions. The transformation of one design issue into another is defined as a design process (Gero, 2010). We define a design idea in design ideation using a co-creative system as a cognitive issue.

To measure ideation in a co-creative system, we developed three metrics based on (Shah et al., 2003), used for evaluating idea generation in design: novelty, variety, and quantity of design. We define the effect of the co-creative system as contributions of the AI agent to the idea generation. Two basic criteria are identified to define the contributions of the AI agent based on (Shah et al., 2003):

- How well does the AI agent contribute to expanding the design space?
- How well does the AI agent contribute to exploring the design space?
We employ two approaches with the three metrics: an aggregated approach and a temporal approach. The aggregated approach allows us to evaluate the contributions of an AI agent in a design ideation. The representation of ideation process by temporal changes of ideas allows to (1) compare an ideation process of a design session to other design sessions, (2) identify specific patterns of novelty, variety, and quantity of ideas in a condition, (3) identify specific contributions of the co-creative system associated with novelty, variety, and/or quantity.

Aggregate Analysis

For the aggregate approach, we developed three metrics based on (Shah et al., 2003), used for evaluating idea generation in design: novelty, variety, and quantity of design. Novelty is a measure of how unusual or unexpected an idea is as compared to other ideas (Shah et al., 2003). A novel idea is defined as a unique idea across all design sessions in a condition. For measuring novelty, we count how many novel ideas in the entire collection of ideas in a design session then divide the novel ideas by the number of AI contributions (e.g., inspiring sketches and images that the AI agent provides) that the designer gets from a co-creative system, as shown in Equation (1). The novelty score thus means the number of novel ideas per AI contribution in a design session. Variety is a measure of the explored solution space during the idea generation process (Shah et al., 2003). The generation of similar ideas indicates low variety and hence, less probability of finding better ideas in other areas of the solution space. For measuring variety, we code each idea whether it is a new idea or a repeated idea in a design session and only the number of new ideas is counted in a design session then divide the new ideas by the number of AI contributions that the designer gets from a co-creative system, as shown in Equation (2). Quantity is the total number of ideas generated, generating more ideas increases the possibility of better ideas (Shah et al., 2003). For measuring quantity, the number of ideas both new ideas and repeated ideas is counted in a design while the metric of variety includes only new ideas, as shown in Equation (3). These metrics enable a comparison of a designer’s exploration of design space while using different AI models in ideation.

1. Novelty Score:
   \[
   \frac{\sum \text{(Unique ideas across all design sessions in a condition)}}{\sum \text{(AI contributions in a design session)}}
   \]

2. Variety Score:
   \[
   \frac{\sum \text{(New ideas in a design session)}}{\sum \text{(AI contributions in a design session)}}
   \]

3. Quantity Score:
   \[
   \frac{\sum \text{(New ideas in a design session)} + \sum \text{(Repeated ideas in a design session)}}{\sum \text{(AI contributions in a design session)}}
   \]

Temporal Analysis

A single value for each of novelty, variety, and quantity of ideas in a design session enables of measure of ideation to be associated with the entire design session. We include a temporal analysis of ideation to enable a characterization of the flow of ideas during a design session. We divide a design session as a series of segments bounded by the input of inspiration from the AI agent. For the temporal analysis, the number of novel ideas, the variety of ideas, and the quantity of ideas are calculated for each segment to produce a sequence of temporally ordered ideas in a design session. The nuances of the ideation process are then illustrated by temporal changes of novelty, variety, and quantity of ideas over time.

![Example of temporal analysis](image)

Conclusion

Measuring ideation when co-creating with an AI-based co-creative design tool enables the comparison and evaluation of the impact of different AI models on the user’s cognitive process and the creative outcome. In order to measure ideation, we developed an approach for measuring ideation that has two components: an aggregate analysis and a temporal analysis. The aggregate analysis adapts existing quantitative metrics for ideation: novelty, variety, and quantity of ideas expressed in the design session. The temporal analysis shows the temporal changes of novelty, variety, and quantity of ideas based on the AI contributions. These measures can be used in evaluating the impact of AI contributions in other co-creative systems that support design creativity. We conclude that the approach that we developed for measuring ideation can allow different AI models for inspiration to be compared and justified.

References


COFI: A Framework for Modeling Interaction in Human-AI Co-Creative Systems

Jeba Rezwana, Mary Lou Maher
Software and Information Systems
University of North Carolina at Charlotte
9201 University City Blvd, Charlotte, NC 28223, US
jrezwana@uncc.edu, m.maher@uncc.edu

Abstract

Human-AI co-creativity involves both humans and AI collaborating on a shared creative product as partners. In a creative collaboration, interaction dynamics, such as turn-taking, contribution type, and communication, are the driving forces of the co-creative process. Therefore an interaction model is an essential component for designing effective co-creative systems. There is relatively little research about interaction design in the co-creativity field, which is reflected in a lack of focus on interaction design in many existing co-creative systems. This paper focuses on the importance of interaction design in co-creative systems with the development of the Co-Creative Framework for Interaction design (COFI) that describes the broad scope of possibilities for interaction design in co-creative systems. Researchers can use COFI for modeling interaction in co-creative systems by exploring the possible spaces of interaction.

Introduction

In human-AI co-creative systems, humans and AIs collaborate in a creative process as creative colleagues, and the focus is on co-creative partnerships in contrast to creativity support tools (Davis 2013). Creative collaboration involves interaction among collaborators, and the shared creative product is more than each individual alone could achieve (Sawyer and DeZutter 2009). Sonnenburg demonstrated communication as the driving force of collaborative creativity (Sonnenberg 1991). Interaction is a basic part of co-creative systems as both the human and the AI actively participate and interact with each other. Designing co-creative systems have many challenges due to the open-ended nature of the interaction between the human and AI (Davis et al. 2016). Bown asserted that the success of a creative system’s collaborative role should be further investigated concerning interaction design as interaction plays a key role in the creative process (Bown 2015).

Interaction design is often an untended topic in the co-creativity literature despite being a fundamental property of co-creative systems. In recent years, researchers designed many co-creative systems that are very intriguing and creative, yet sometimes users fail to maintain their interest and engagement while collaborating with the AI due to the unimpressive quality of collaboration. An adequate interaction model dramatically improves the quality of the collaboration and usability of a system. Therefore, as a young field, a holistic framework that captures the scope of interaction design is necessary. A good starting point to investigate questions about interaction modeling is studying creative collaboration in humans (Davis et al. 2015). Understanding the factors of human collaboration can be a tool to build the foundation for the development of systems that can augment or enhance creativity in humans (Mamykina, Candy, and Edmonds 2002). The literature on computational creativity and computer-supported collaborative work (CSCW) can also help identify interaction components related to human-AI co-creativity.

In this paper, we present the Co-Creative Framework for Interaction design (COFI) that defines interaction components in a co-creation to describe the broad scope of possibilities for interaction design in co-creative systems. These interaction components represent various aspects of a co-creation, such as participation style, contribution type, and communication between humans and the AI. COFI is informed by the literature on human collaboration, CSCW, computational creativity and human-computer co-creativity. We adopted interaction components based on a literature review and adapted the components to concepts relevant to co-creativity. We argue that COFI can be used as a guide when modeling interaction in co-creative systems as researchers can use COFI for exploring the possible spaces of interaction. COFI can also be useful while analyzing and interpreting the interaction design of existing co-creative systems.

Related Work

In co-creative systems, humans and AI both contribute as creative colleagues in the creative process (Davis 2013). Creativity that emerges from human-AI interaction cannot be credited either to the human or to the AI alone and surpasses both contributors’ original intentions as novel ideas arise in the process (Liapis, Yannakakis, and Togelius 2014). Designing interaction in co-creative systems has unique challenges due to the spontaneity of the interaction between the human and the AI (Davis et al. 2016). A co-creative AI agent needs continual adjustment and adaptation to cope with human strategies. Mamykina et al. argued that by understanding the factors of collaborative creativity among humans, methods can be devised to build the foundation for the...
development of systems that can augment or enhance collaborative creativity (Mamykina, Candy, and Edmonds 2002).

In the field of co-creativity, interaction design includes various parts and pieces of the interaction dynamics between users and the AI, such as participation style, communication, contribution type etc. Bown argued that the most practiced form of evaluating creative systems is mostly theoretical and not empirically well-grounded and suggested interaction design as a way to ground empirical evaluations of computational creativity (Bown 2014). Yee-King and d’Inverno also suggested a need for integration of interaction design practice into co-creativity research (Yee-King and d’Inverno 2016). There is a lack of a framework for interaction design, which is necessary to explain and explore the possible interaction spaces and compare and evaluate the interaction design of existing co-creative systems to improve the practice of interaction modeling.

Interaction among the individuals in collaboration makes the process emergent and complex. For investigating human collaboration, many researchers stressed the importance of understanding the process of interaction. Fantasia et al. proposed an embodied approach of collaboration that considers collaboration as a property and intrinsic part of interaction processes (Fantasia, De Jaeger, and Facsulo 2014). Schmidt defined CSCW as an endeavor to understand the nature and characteristics of human collaboration to design adequate computer-based collaborative technologies (Schmidt 2008).

Co-Creative Framework for Interaction Design (COFI)

We develop and present Co-Creative Framework for Interaction design (COFI) as a guiding tool that describes the broad scope of possible spaces for interaction design in co-creative systems. This framework describes various aspects involved in the interaction between the human and the AI. COFI is informed by research on human collaboration, CSCW, computational creativity, and human-computer co-creativity.

The primary categories of COFI are based on two types of interactional sensemaking of collaboration as described by Kellas and Trees (Kellas and Trees 2005): interaction between collaborators and interaction with the shared product. Interaction with the shared product, in the context of co-creative systems, describes interaction aspects related to the creation of the creative content. Interaction between collaborators explains how the interaction between the human and the AI is unfolding through time which includes turn-taking, roles, timing of initiative, communication, etc. Thus, COFI characterizes relational interaction dynamics between the collaborators (human and AI) as well as functional aspects of interacting with the shared creative product. We choose the Kellas and Trees framework for the primary categories of COFI because they used their framework as a tool for explaining and evaluating the interaction dynamics in human creative collaboration in joint storytelling.

As shown in Figure 1, we further divide the two primary categories of COFI, interaction with collaborators and interaction with the shared product, into four subcategories. Interaction between collaborators is divided into collaboration style and communication style, inspired by research in CSCW and HCI. Interaction with the shared product is divided into the creative process and creative product, inspired by research in creativity and, more specifically, computational creativity. CSCW literature discusses collaboration mechanics among humans to make effective CSCW systems, whereas creativity literature talks about creative process and product. The interaction components are child categories of the four main subcategories and are adapted to the context of human-AI co-creativity. COFI is not a complete ontology and can continue to expand as new interaction components emerge.

Interaction between Collaborators (Human and AI)

This section presents components related to the relational interaction dynamics between the human and the AI as co-creators. As shown in Figure 1(a), interaction between collaborators is divided into two subcategories which are collaboration style and communication style.

Collaboration Style

Collaboration style is about different parts and pieces of interaction between humans and the AI, related to the nature of the co-creation. The following subsections describe each interaction component in this category.

Participation Style: Participation style in COFI refers to whether the collaborators can participate and contribute simultaneously, or one collaborator has to wait until the partner finishes a turn. Therefore, participation style in COFI is categorized as parallel and turn-taking. Categorization of participation stems from works on cooperation in the literature (Johnson and Johnson 2005; Liu, Saito, and Oi 2015).

Task Distribution: Task distribution refers to the distribution of tasks among the collaborators in a co-creative system. In COFI, there are two types of task distribution, same task and task divided. When it is same task, there is no division of tasks among collaborators and the collaborators take part in the same task. For example, in a human-AI co-creative drawing, both co-creators do the same task, generating the drawing. In a task-divided distribution, the main task is divided into specific sub-tasks and the sub-tasks are distributed among the collaborators. For example, in co-creative poetry, the user can generate a poem while the AI agent can evaluate the poetry. This component of COFI emerged from discussions of the two interaction modes presented by Kantasalo and Toivonen: alternating and task divided co-creativity (Kantasalo and Toivonen 2016).

Timing of Initiative: In a co-creative setting, the timing of both parties’ initiative taking can be scheduled beforehand, or it can happen naturally in real-time. If the timing of the initiative is fixed in advance, in COFI, it will be addressed as planned. If collaborators initiate their contribution naturally without any prior plan or fixed rules, it will be addressed as spontaneous. Timing of initiative should be chosen based on the motivation. Spontaneous timing is suitable for increased emergent results, whereas planned timing is suitable for systems where users want inspiration or help in a specific way for a particular aspect. Salvador et al. discussed tim-
Figure 1: Co-Creative Framework for Interaction Design (COFI): On the left (a) Components of Interaction between the collaborators. On the right (b) Components of Interaction with the Shared Product.

The Co-Creative Framework for Interaction Design (COFI) is a model for evaluating groupware for supporting collaboration (Salvador, Scholtz, and Larson 1996).

**Communication Style** Communication is a vital component in any collaboration for the co-regulation between the collaborators and helps the AI agent make the proper decision in a creative process. Communication style includes different kinds of channels to communicate between users and the AI.

**Human to AI Intentional Communication** Human to AI intentional communication channels represent the possible ways a human agent can intentionally communicate to the AI agent to provide feedback and convey important information to each other. Gutwin and Greenburg proposed a framework for groupware that discusses the mechanics of collaboration and it includes intentional communication as a major element of collaboration mechanics (Gutwin, Greenberg, and Roseman 1996). In COFI, human to AI communication channel includes direct manipulation, voice, text and embodied communication. The user can directly manipulate the co-creative system by clicking buttons for giving instructions or feedback or inputs and providing user preferences by selecting from AI-provided options. Using the whole body or gestures for communicating with the computer will be referred to as embodied. According to HCI modalities, intentional communication from human to AI includes direct manipulation, gesture, text, and voice (Nigay 2004).

**Human to AI Consequential Communication** In COFI, human to AI consequential communication channels represent the ways a co-creative AI agent can track and collect unintentional or consequential information from the human user, such as eye-tracking, facial expression tracking and embodied movements. Tracking consequential details from the human is essential to perceive user preference, user agency and engagement. Gutwin and Greenburg reported consequential or unintentional communication as a major element of collaboration mechanics, in addition to intentional communication (Gutwin, Greenberg, and Roseman 1996).

**AI to Human Communication** Humans expect feedback and evaluation of their contribution from collaborators. Therefore, if the AI agent can communicate their status and feedback for a contribution, it would make the co-creation more balanced as the AI agent will be perceived as an equal partner rather than a mere tool. The AI to user communication can include text, voice, visuals (icons, image, animation), haptic and embodied communication according to HCI modalities (Nigay 2004).
Interaction with the Shared Product

Interaction components related to the shared creative product in a co-creative setting are discussed in this section and illustrated in Figure 1(b). Interaction with the shared product is divided into two subcategories, creative contribution to the product and creative process.

Creative Process Creative process characterizes the sequence of actions that lead to a novel, and creative production (Lubart 2001). In COFI, there are three types of creative processes that describe the interaction with the shared product: generation, evaluation, and definition. During a creative generation, creative artifacts and ideas are produced in a specific conceptual description. In a creative evaluation, produced ideas, artifacts or concepts get assessed to be more refined and appropriate for the creative objective. In a creative definition process, collaborators determine and prepare the creative conceptual space. For example, a co-creative AI agent can define the attributes of a fictional character before a writer starts to write about the character. The basis of this categorization is the work of Kantosalo et al. that defines the roles of the AI as generator, evaluator, and concept definer (Kantosalo and Toivonen 2016).

Creative Product The creative product is the idea or concept that is being created. Creative product has two interaction components, contribution type and contribution similarity. (Lubart 2001). Contribution Type: In a co-creation, an individual can contribute in different ways to the shared product. Co-creators can generate new elements, extend the existing contribution, modify or refine the existing contribution. The primary contribution types in COFI are: ‘create new’, ‘extend’, ‘transform’ and ‘refine’. ‘Extend’ refers to extending the contribution of the partner or adding on to the partner’s contribution. Generating something new or creating new objects is represented by ‘create new’, whereas ‘transform’ conveys turning the partner’s contribution into something totally different. ‘Refine’ is evaluating and correcting the partner’s contributions with a similar type of contribution. Contribution types are adopted and adapted from Boden’s categories of computational creativity based on different types of contribution: combinatorial, exploratory, and transformational (Boden 1998).

Contribution Similarity: Contribution similarity refers to the degree of similarity or association in terms of the contribution compared to the partner’s contribution. Both convergent and divergent exploration have value in a creative process. Basadur et al. asserted that divergent thinking is related to the ideation phase and convergent thinking is related to the evaluation phase (Basadur and Hausdorf 1996).

Conclusions

As a growing field, human-computer co-creativity lacks significant research on interaction models and their implications in co-creative systems. Human-AI co-creativity research needs a holistic framework that captures aspects and components of interaction to design effective co-creative systems. In recent years, a few frameworks have been developed about interaction design in co-creative systems. However, they lack a focus on interaction components related to the interaction between collaborators as distinct from interaction components related to the shared product. We develop and describe COFI as a new framework to provide a way for researchers to explore the design space of interaction for a specific system. COFI will provide useful guidelines for interaction modeling while developing co-creative systems. COFI can also be beneficial while investigating and interpreting the interaction design of existing co-creative systems. As a framework, COFI is expandable as other interaction components can be added to it in the future. By establishing COFI, we can look for the relationship between different interaction designs in co-creative systems and different creative outcomes. COFI can be also used to develop co-creative systems that can improve user engagement with effective collaboration strategies through adequate human-AI interaction.

References


Davis, N. M. 2013. Human-computer co-creativity: Blending human and computational creativity. In Ninth Artificial Intelligence and Interactive Digital Entertainment Conference.


Liapis, A.; Yannakakis, G. N.; and Togelius, J. 2014. Computational game creativity. ICCC.


Yee-King, M., and d’Inverno, M. 2016. Experience driven design of creative systems.
Agency in Co-Creativity: Towards a Structured Analysis of a Concept

Janin Koch
ExSitu Lab
Université Paris-Saclay-CNRS,Inria,LISN
Paris, France
Janin.Koch@inria.fr

Prashanth Thattai Ravikumar
Communications and New Media
National University of Singapore
Singapore
prashanth.thattai@gmail.com

Filipe Calegario
Mustic, Centro de Informática
Universidade Federal de Pernambuco
Brazil
fcac@cin.ufpe.br

Abstract
Computational co-creativity is an emerging area of research in co-creativity, that connects to user-centered research in design, human-computer interaction and artificial intelligence. A common vocabulary among researchers is critical for advancing research in such interdisciplinary work, but it is often difficult to achieve. The concept of agency is one of the topics of interest for the development of computational co-creative systems. While agency can be broadly defined as a sense of control over the creative outcomes of a co-creative process, there is no shared understanding of how to design systems for and with agency. In this paper, we reflect on the works we have developed over the last five years as a starting point for discussing the definition of agency. Further research will allow us to investigate how agency manifests itself in human-computer creative collaboration, also known as co-creativity. We believe that having a common ground allows for a more efficient process of analyzing and designing human-computer collaborative systems.

Introduction
Interdisciplinary domains, such as computational creativity (CC), frequently present difficulties for researchers in finding common ground between observed phenomena and formalizing concepts in a shared manner. Co-creative research faces an even greater challenge with computational co-creativity, as it has a greater degree of overlap with domains of user-centered research in design and HCI.

One of the challenges in this emerging area of co-creativity research is developing a common vocabulary that will allow researchers to better communicate their accomplishments in various domains of co-creativity. To that end, we propose developing a common vocabulary centered on the concept of agency in order to analyze and synthesize various strands of co-creativity research.

Our research group’s necessity led to a better definition of the concept of agency. The group comprises of three researchers from three continents (South America, Europe, and Asia), each with our own set of experiences and research interests. Our concepts, practices, and processes can diverge significantly when working in multidisciplinary environments. However, to collaborate successfully in such an environment, a shared vocabulary is required to find common ground. To overcome such differences, we saw the need to take a step back and better understand the concepts before applying them to creating a collaborative body of knowledge in human-computer co-creativity.

We further began a search for uses of agency in ICCC proceedings in the literature. Based on this work-in-progress analysis, we identify a few factors in current definitions of agency that we believe are influential to the agency of co-creative systems and would like to discuss with workshop participants. Identifying and designing systems with these factors in mind will assist researchers in developing co-creative applications and communicating their work to others.

Developing Co-creative Systems with Agency
Our individual research interests range from human-to-human interaction to human-machine collaboration in a variety of domains. While not exhaustive, this review does provide insights into the role agency may play in the design, evaluation, and reflection of co-creative systems.

Co-creative systems in human-human co-creative process
The third author discusses the human-human co-creation process of prototyping digital musical instruments (Calegario 2019). Musicians and designers used a modular prototyping toolkit that encapsulates technical details and accelerates experimentation.

The Probatio system comprises bases and blocks, which aims to bridge the gap between abstract concepts and working prototypes by making it simple to experiment with postures and gestures that generate sound in real-time.

Besides the individual interaction with the system, designers and musicians can cooperate with Probatio, physically communicating intentions, presenting and experimenting with blocks that can be used as sensors for particular intentions and circumstances. The artifact becomes a mediator between two human creative agents, making it straightforward for the group to impact each individual in the final creative result. In that sense, the concept of agency relates to feeling in control of the instrument’s design choices and the means to produce music.

1https://probat.io
The agency concept in Probatio relates to the term *material agency* discussed by Brown (Brown 2016), who building upon the work of Malafouris, combines the human intention with the material affordance of the artifact (Malafouris 2008).

In addition to the perceptible material possibilities, the responsive nature of the system allows for the agency in the creation process of a digital musical instrument. The system's encapsulation of technical details opens possibilities for people with little technical experience to see themselves in constructing digital artifacts, clearly perceiving the effect of their actions in creating digital musical instruments.

**Computational co-creative systems for creative collaboration**

Through experiments with a system developed for real-time rhythmic duets, the second author highlights the effect of agency on user experiences with a music co-creation system. The MASSE system was developed for real-time rhythmic duets and employs a combination of high-level constraints and stochastic rhythm generation to select its rhythmic behaviors (Ravikumar and Wyse 2019). MASSE was evaluated by creating different configurations of the system that played duets with the musicians. The different configurations produced musical responses with different degrees of complementary in its rhythmic support (Ravikumar and Patel 2020). Musicians reported a satisfactory sense of co-creative engagement different from controlling a tool when the system produced suggestive rhythms and played rhythms that opposed their musical ideas. In configurations that the system behaved predictably and mirrored their contributions, musicians felt that the system was a good follower. Musicians perceived the system as least co-creative when it generated rhythms that were not correlated with their contributions.

The development and experimentation of the MASSE system augments the current work on agency in music co-creation systems. In prior work, humans have perceived agency in music co-creation systems that exhibit autonomy (Bown 2018), musical personality (Albert 2013), initiative through unpredictable behaviors (Lewis 2000), and supportive yet interactive behaviors (Brown, Gifford, and Voltz 2016). Experiments with the MASSE system suggest that complementary behaviors can have predictable effects of user experiences of co-creativity (Ravikumar and Patel 2020). This is, however, a preliminary result and needs to be examined in greater detail with subsequent experiments.

**Human-centered approaches on human-AI collaboration**

Designing collaborative systems with a human-centric perspective requires developers taking into consideration the impact of systems on both the creative process and the practitioners themselves. Author one investigated how different design aspects of collaborative systems result in diverse perceptions of agency within a creative process through various studies.

The *MayAI?* system was designed to create a one-to-one situation between a designer and a system in order to find and develop ideas visually in the form of an image collage (Koch et al. 2019). The system was able to explore and exploit visual ideas by suggesting images that were either similar to or diverged from the current collage in order to inspire new ideas or even provoke the designer. The system then inquired about the designer’s reasoning for selecting specific images that differed from the current collage. As a result, designers attributed agency to the system in the form of authorship (“I feel like I don’t work alone, I feel like there is another person” (P15)) or having its own agenda (“I think it was a ‘she,’ and she maybe heard me but she had her own opinions as well” (P14)). Such remarks were primarily attributed to the system’s autonomy to follow and explore specific ideas. This is consistent with Bown’s understanding that autonomy encourages users to perceive agency (Bown 2018). Interestingly, the system’s ability to evaluate and request reasoning from participants was viewed as beneficial for reflection on current practice, but also sometimes as a criticism of their choices. This demonstrates the potential of interpretive agency in creative practice, as proposed by Fischer et al. In (Fisher and Shin 2019), they argue that in order to fully understand and model creative agency, we have to extend our view from generative agency, the act of creating objects, to computationally-modelled interpretive agency, the evaluation of the creative value. This would enable systems to contribute narrative critique, criticism, and commentary, in order to have more agency about their creations and contributions. However, our results also highlight the risk computational interpretation can pose to the creator’s self-perception, potentially leading to negative experiences or rejection of collaboration.

Furthermore, the *ImageSense* system was created to investigate the role and perception of intelligent systems in the creative process (Koch et al. 2020). It incorporates five different intelligent tools with varying degrees of agency into a visual ideation system in which multiple designers can create and discuss ideas using a visual collage. In this context, agency can be defined in terms of two dimensions: pro-activity and adaptability. The former refers to how much it interacts with and makes its own suggestions, similar to (Lewis 2000; Brown, Gifford, and Voltz 2016), and the latter to how much the system’s contributions adapt to the current creative process. In studies with eight design teams, the perceived level of agency increased with both dimensions. The role of the included tools in the process was defined by this perception. Systems that primarily adapted to the designer’s actions were perceived as a design function, whereas systems that made their own suggestions were recognized as assisting the creative process. When both dimensions were present, the system was perceived as a potential partner who could take the lead if the designer became stuck on an idea. This preliminary finding will need to be investigated further in future studies that include more and different dimensions.

**Discussion**

A common thread for furthering agency research in co-creative systems emerged from the interactions between the authors. The observations from co-creative systems for music and creative ideation suggest that pro-activity and adapt-
ability are desirable characteristics for an interactive system to be perceived as a co-creator. Furthermore, observations of human-human co-creation with the Probatio system point in the direction of future research on agency in co-creative systems via the design of co-creative spaces.

Agency between pro-activity and adaptability

While some of the approaches described above differ from one another and are adapted to the type and domain of application, we can see recurring themes that influence system design and agency perception. All of our work tries to investigate the user’s perception of agency, how it emerges, and to begin to understand how certain behavioral characteristics influence it. One topic that emerged in our discussions was the trade-off we see between pro-activity/autonomy and adaptability. According to the results of our individual experiments, both factors contribute to a user’s perception of agency. However, the individual dimensions do not appear to be sufficient to elicit a strong sense of agency.

In ImageSense, systems that actively suggested images that were not directly related to the ideation process were frequently regarded as beneficial to the creative process, but did not convey a sense of agency that made them appear co-creative despite their active contributions. Similar observations were reported by musicians who played with the configuration of MASSE that produced uncorrelated rhythms. Multiple musicians acknowledged the system’s ability to generate independent rhythms. However, they did not feel as co-creative as compared to the conditions in which the system generated related material.

On the other hand, adaptability, or the ability to adjust to new conditions, enabled the system in staying relevant to the current task. In ImageSense, however, systems that adapted closely to the designer’s actions were perceived as design tools that helped to reflect and clarify current activities rather than active co-creators. Reports from MASSE were also consistent with this observation. In configurations that system behaved predictably and mirrored the musician’s contribution, musicians felt that they were more in control of the performance and considered the system as a good follower. However, they did report that their interaction with the system was one of performing with a co-creator.

The above mentioned observations lend strength to the idea that humans may perceive interactive tools as co-creators when they exhibit a balance between pro-activity and adaptability. Such systems exhibit an ability to generate relevant contributions that lie on a range between exploiting (following) the current task closely and exploring (diverging) from it to some extent. Among our different works, the complementary behaviors developed in the MASSE system is one such example. While complementary rhythms oppose the musician, they also maintain relatedness to the inputs. Preliminary experiments with the system suggest that complementary responses have a positive on co-creative experiences. The relation between complementary behavior and agency is a connection that needs to be examined in greater detail with subsequent experiments.

Agency and design space

Co-creative systems, such as the aforementioned, are frequently designed with a predefined design and decision space in which the system can navigate and make autonomous choices.

The Probatio system, on the other hand, explores a scenario in which one or more of the creative agents co-creates the design space itself. The concept of agency in this context refers to a sense of control over design choices (e.g., the instrument’s design choices). However, extending and modifying the design space generally requires a more holistic understanding of the task and context, which is a challenge for current systems. We anticipate that computational co-creative systems that co-create the design space will become a viable approach to increase agency in the future.

Beyond Individual Experiences with Agency

Our work aims to build a shared vocabulary for long-term design of the agency of co-creative systems. This paper focuses on a small brick that provides some hands-on examples of approaches to develop systems with agency. We believe that exercises like this can help us get closer to a more precise definition of the term agency and the various concepts that surround it. We have already begun to broaden our investigation in this context, and we believe that this collaborative research on agency will benefit the larger community in two ways: a better understanding of the impact of agency in practice, and a more general model of agency for developing more advanced co-creative systems.

Understand agency in practice

The first goal of this work is to better understand from researchers what agency means in their work and how they use it in system development. Going forward, this entails generating discussions and developing questions that allow authors to articulate their systems from the standpoint of agency. We would like to pose the following questions to the researcher community in order for them to critically think about their system design and practice from the standpoint of agency:

- What do we mean by agency in computational creativity?
- What are the dimensions of agency?
- What are alternative words/concepts that people use in relation to agency?
- What effect does agency have on human-computer creative collaboration, i.e. decision making or leadership?
- How does agency manifest itself in human-computer creative collaboration, for example, in terms of control, steerability, or authorship?
- How does the definition of agency support us in developing co-creative projects?
- How could you convey agency in the system, for example, through goals or autonomy?

Based on our findings, there seems to be a disconnect between theoretical work on agency and practical implementation of agency in systems, which commonly refer to other
existing concepts. Collecting more examples of co-creative systems designed with agency would not only broaden our understanding of agency, but would also contribute to closing the gap, resulting in a more holistic definition of agency.

**Develop a framework of agency**

The second goal of this work is to create a framework for using agency in system design and analysis.

To that end, we have begun to examine the definitions of agency in eleven editions of the proceedings of the International Conference on Computational Creativity (2010 to 2020). There are 513 papers in total, including full articles, short articles, demo papers, and position papers. First, we used Atlas.TI software to search the ICCC corpus for the string agency. Then we eliminated the papers in which the word differed from our researched theme, such as agency as institution or organization. Then, based on word frequency, we chose papers that contained the phrase agency at least five times. Finally, we came up with a shortlist of twelve papers that we divided among the research team for further examination.

Throughout these papers, one common observation was that agency was either attributed to the system itself, referring to specific functionalities, or to the perceived agency of the system by the users. Examples of the former include system intentionality (Guckelsberger, Salge, and Colton 2017) and decision making process (Gemeinboeck and Saunders 2013). The latter frequently referred to aspects such as perceived agency in movements (Saunders and Gemeinboeck 2018) or the system’s ability to provide alternatives that the user can evaluate (Mateas, Mawhorter, and Wardrip-Fruin 2015). We intend to further investigate this divide in our future work. As the number of papers that directly refer to agency is rather limited we will also extend our scope to papers describing agency using related wordings and concepts.

**Conclusion**

Developing a common vocabulary around the notion of agency for designing co-creative systems is a crucial matter. It will benefit the collaboration of inter-disciplinary research teams to advance co-creative systems. In order to achieve this we need to explore how agency is understood in practice by the community as well as in theory. Finally, more research is needed to identify the relevant dimensions concepts that will enable a clearer definition of agency – how to define it, measure it, and communicate it to others.

**References**

Albert, J. 2013. Interactive musical partner: A demonstration of musical personality settings for influencing the behavior of an interactive musical generation system. In *In Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*.


AI-Aided Co-Creation for Wellbeing

Haizi Yu¹, James A. Evans¹, Donna Gallo², Adam J. Kruse², William M. Patterson², and Lav R. Varshney²

¹ University of Chicago, IL 60637, USA  {haiziyu, jevans}@uchicago.edu
² University of Illinois at Urbana-Champaign, IL 61801, USA  {dogallo, akruse, wmpatter, varshney}@illinois.edu

Abstract

Human health is a state of complete physical, mental, and social wellbeing; not merely absence of disease or infirmity. Longstanding research in the psychology of wellbeing lists mastery, relatedness, and autonomy as the three innate psychological needs that must be fulfilled. Group creativity, when carefully designed and coordinated, is a unique activity that fulfills all three simultaneously, boosting not only global wellbeing but also cultural and economic wealth of diverse communities. However, group creativity is difficult and can thwart competence if not done properly: individual creative contributions are fundamentally complex and co-dependent; their combination requires more intelligence than simple summation or independent voting. We propose a vision to advance AI-aided social co-creation platforms that boost interactive and effective human-human and human-AI collaborations in creating music, which can then expand to other creative and productive domains. Current videoconferencing, distance learning, and social networking helped maintain connection in these pandemic years, but are insufficient for people to collaboratively create. We leverage a new human-interpretable learning paradigm—information lattice learning—together with equity-centered design principles, to build AI-aided platforms that integrate disparate groups of people, their ideas, and expertise into high-quality creative outputs while preserving autonomy of individual human contributors and supporting anti-racist teaching efforts.

Introduction

In the preamble to its constitution, the World Health Organization states that health is a state of complete physical, mental, and social wellbeing and not merely the absence of disease or infirmity. Many believe that even with COVID-19 abating in many communities, our lives are going to be more physically constrained and more virtual than before (Ramnastu-Avalos and Siiman 2020). Surgeon General Vivek Murthy and others have warned that social distance will cause a crisis of loneliness and a social recession with permanent damage to communal bonds that are essential to health and wellbeing (Murthy and Chan 2020; Murthy 2020). This position paper argues that it is possible to improve human wellbeing and therefore health via a platform for co-creative music composition with anyone in the world, powered and coordinated by artificial intelligence (AI). Moreover, it can support critically creative musical collaborations that build cross-cultural competence and foster social change in communities by developing knowledge of self among educators, students, and community members (an element of Hip Hop that is key to transformative practice, goals of racial justice, and development of critical consciousness).

In particular, we argue such a co-creativity platform can be built on emerging foundational AI theory on interpretable learning paradigms (e.g., information lattice learning) (Yu and Varshney 2017), human-AI interaction design, music intelligence, sociology of creative teams (Yin, Wang, Evans, and Wang 2019; Wu, Wang, and Evans 2019), computational (co-)creativity for mental wellbeing (Cheatley, Moncur, and Pease 2019; Cheatley, Ackerman, Pease, and Moncur 2020), street inquiry methods for social impact in research, Hip Hop entrepreneurship and the commercialization of AI creativity technology, as well as relevant lived experience. We further argue such a platform built on interdisciplinary foundations will enable economic and societal impact; answer significant research questions at the intersection of artificial intelligence, the arts, and the social sciences; and grow the cultural and economic wealth of diverse social groups, e.g., Black communities in the United States.

Designing AI-aided social co-creation platforms can boost interactive and effective human-human and human-AI collaborations in creating art and science. However, in addition to the underlying AI technology, the user interface and user experience (UI/UX) must also be high-quality to ensure human creators and AI can initiate and maintain effective interaction. Moreover, large-scale adoption of a consumer-facing technology requires it to have broad appeal.

We argue the following pipeline of key activities for human-AI interaction-focused research and development will yield a music co-creation platform.

1. Develop further theory on collective human and artificial intelligence.
2. Design practical UI/UX to support deep human-human and human-AI creative interaction, leading to equity-centered music composition.
3. Implement real AI-aided platforms and explore their potential for economic/societal impact.
4. Work with diverse user groups (e.g., K-12 classrooms, Black communities) to inform tech features and human-AI interaction via user-centered design, and to support critically creative musical collaborations that build cross-cultural competence and foster social change in communities by developing knowledge of self.

Besides developing AI agents as team members working with people, it is also important to design AI “admins” coordinating teamwork and capturing value from individual contributors. We leverage equity-centered methodology as our design principle for such AIs: it is the practice of purposefully involving diverse communities (including minority social groups) throughout a design process so as to allow all voices to be directly heard and to directly affect how the solution will address the inequity at hand. It is a form of user-centered design (Vredenburg, Mao, Smith, and Carey 2002; Garrett 2010; Endsley 2016), an iterative design process in which designers and system-builders focus on users and their needs in each phase of the design and build process.

The co-creative future between humans and AIs

Creativity is powerful. Regarded as one of our most sophisticated cognitive skills, this ability drives human progress by allowing us to perform nonroutine tasks, take advantage of novel opportunities, and invent new solutions to problems facing the world. Creativity is the hallmark of art and science, as well as engineering and technology that benefits wide swaths of society. More broadly, it is the basis for innovation and continuous reinvention—enhancing creativity can accelerate and improve product and service development across all industries.

Co-Creativity is critical. Although popular culture tends to lionize the lone genius, group creativity often trumps individual creativity. It is not surprising that creative work is largely carried out by teams rather than heroic solo inventors. Teamwork, however, comes with its own dynamics. Effective collaboration requires not only cooperation, i.e., having aligned goals, but also coordination mechanisms to enable effective alignment and adjustment to teammates’ actions. Both are enhanced in social creativity if teammates have an effective theory of mind, or the ability to accurately attribute mental states such as beliefs, desires, and emotions to oneself and to others. Proper social workflows and organizational architectures also help. Group music composition is an important example because music is engaging and central to human self-expression and culture; Google’s Bach doodle for music harmonization, only briefly online, catalyzed 350 person-years of creative human engagement.

AI-aided co-creativity is the future. The COVID-19 pandemic restricted co-creativity, with technical, scientific, engineering, musical, and other creative teams largely shifted to remote work. Videoconferencing, social networking, digital curation, and education tools have all seen significant increases in usage as teams interact virtually. Unfortunately, these tools seem to be ineffective at fully recreating the wide range of social heuristics and institutions that support creativity in shared physical space. These shortcomings have a deleterious effect on both productivity and work satisfaction. AI-aided systems can bridge this disconnect and help effectively rebuild what has been lost during the pandemic. Although started as a solution, AI-aided co-creation may in turn trigger a new revolution that fundamentally changes how intelligences, natural and artificial, interact in the long run.

Enhancing wellbeing

In psychology, self-determination theory states that people across cultures manifest three innate psychological needs: mastery, relatedness, and autonomy (Ryan and Deci 2000; Varshney 2012; Church et al. 2013). Satisfying these needs is said to lead to psychological health and wellbeing, which in turn leads to greater creativity, effective problem solving, motivation, performance, and persistence. While existing remote tools often satisfy one or two of these needs, group creativity activities satisfy all three at once, yielding the highest levels of health and wellbeing. For these benefits to accrue, however, AI-based creativity systems must be able to combine disparate contributions while retaining the distinct creative contributions from each person in order to maintain a sense of autonomy and broad personal identification with the co-created work. In ensemble composition, melodic contributions by people should not be overwhelmed by the harmonization of AI so they cannot be identified and appreciated. Unfortunately, cognitive support tools have a tendency to change content for creative activities and reduce the feeling of human agency, which people indicate is important to feeling ownership of creative artifacts.

AI as a team member: human-interpretable models

How to disaggregate/aggregate disparate contributions in complex tasks such as composing music has been an open question in human-computer interaction (Frich, Vermeulen, Remy, Biskjaer, and Dalsgaard 2019) and collective intelligence (Kittur, Lee, and Kraut 2009). Yet we argue emerging work on human-interpretable and human-like AI (rather than black-box models like neural networks) is up to the challenge, leveraging information lattice learning (ILL) technology first designed for human-interpretable knowledge discovery via information- and group-theoretic
foundations. The technology recovers music theory in the same human-interpretable form as textbooks, as well as discovering powerful new music concepts of interest to music theorists. The basic idea is an iterative discovery algorithm that operates on a generalization of Claude Shannon’s information lattice, which itself encodes a hierarchy of abstractions and grows algorithmically from universal priors (e.g., symmetries) consistent with human innate cognitions.

Of core importance regarding human-AI interaction, the adopted AI approach is both self-exploratory and self-explanatory, so its rule induction and training function can be naturally used for co-creativity. By understanding the basic abstractions underlying contributions from one’s creative partners, one can work better together. Indeed, theory of mind is considered essential for effective human group creativity and even human-AI creativity (Mehta, Somaya, and Varshney 2020). Moreover, methods for taking incoherent sets of rules and making them coherent by violating a minimal set (rather than violating all rules a little bit) (Yu, Li, and Varshney 2017) play an important role in resolving conflicts that typically exist in all sorts of teamwork. Thus, an aggregation function can emerge by exploring, selecting, and re-assembling rules underlying disparate contributions, making them coherent, and then generating novel musical pieces from the new rule set.

The base AI technology is core to develop a music co-creation platform where people and their AI partners can play several roles in a creative ecosystem. Key AI functions include decomposition of music into melody, harmony, rhythm, style, and similar components as well as intelligent synthesis that recombines these into harmonious compositions.

**AI as an administrator: equity-centered design**

Drawing on results from user-centered and equity-centered design methods (supplemented by street inquiry methods) can inform new social theories of human-AI interaction. We develop practical insights for developing UI/UX and networked platforms applicable for diverse communities to capture their values in various creative processes (e.g., young, Black aspiring musicians, producers, and DJs around the world). Notably, interaction design involves:

- Assessing individual strengths (and weaknesses);
- Identifying what might help people with the way they currently do things;
- Exploring what might provide quality user experiences;
- Listening to what people want and getting them involved in design; and
- Using user-centered techniques as part of the design process.

Outcomes of this methodology include flows that describe individual differences and importances in user experience, which in turn inform the technological features needed to support anything anyone wants to do in the system. An example feature that could emerge from a flow would be content-based musical fragment search. As we develop new social theories for the interaction of human and “alien” intelligences and discover the flows of how users actually want to participate in human-AI co-creativity, we expect new technological ideas to emerge to suggest novel AI research questions.

**Assessing wellbeing**

Testing the hypothesis that AI-based music co-creation has significant stickiness, that it enhances mastery, relatedness, and autonomy, and that it indeed leads to increased subjective well-being, is important for any implemented social co-creation platforms. Ethnographic, qualitative and digital information elicitation with users are necessary methodologies. Assessment of specific dimensions of self-determination theory like human autonomy using batteries such as the Index of Autonomous Functioning (Varshney 2020) is another example of effective evaluation.

**Broader societal impacts**

Following our focus on improving global wellbeing via musical co-creativity for everyone, we believe the proposed approach can satisfy aims including: (a) centering Black cultural wealth that connects historical and contemporary musical practices; (b) supporting critically creative musical collaborations that build cross-cultural competence; and (c) fostering social change in communities by developing knowledge of self among educators, students, entrepreneurs/technologists, and community members. In this way, we may address power and structural issues noted in so-called third wave AI ethics. We enable this via the participation of our local partners. One of them is a mobile Hip-Hop classroom and sound studio, a means to collect oral histories, a cross-generational catalyst for music sharing and production, and a method to link communities across the state through music, dance, visual arts, and history. Members of its street team play a crucial role in ensuring participatory, community-driven, culturally authentic experiences. Notably they can identify artists in the community with cultural cachet, and once identified, will onboard them via mindset and skill set development training.

Notably, Hip-Hop is a natural setting for information-lattice based decomposition and recombination. It evolved as an extension of the Black narrative of using music and manipulations of various technologies to speak truth to power, but also pivots beyond current realities with creativity as a form of spiritual survival and flourishing (Petchauer 2012).
Working with AI technology enables honoring and exponentially expanding Black cultural wealth, providing a platform for communicating issues critical to Black communities, and bolstering anti-racist efforts within and beyond our local community, all in an engaging manner that enhances visceral interest in computing technologies.

Acknowledgements
We thank Deron Bell, Michael Manson, Lauren Parks, and Sharon Irish for discussions and comments.

References