Uninformed resource creation for humour simulation

Graeme Ritchie

Computing Science
University of Aberdeen
Aberdeen AB24 3UE, UK.

Abstract: When testing a model of humour generation, it may not be feasible to implement certain crucial modules. We outline how human knowledge manipulation could be used to simulate these modules, without undermining the rigour of the testing process.

Keywords: humour generation, methodology, annotation.

1 Introduction

The area of computer generation of humour [1, Chap. 10] aims to simulate the creation of humorous artefacts (usually written texts), typically via a number of processing stages. If a humour-generating program is needed for some practical purpose, then how it is constructed may be of little theoretical interest. However, in the case of more theoretical research, where the program is there to test an abstract model, the mechanisms are the whole point.

A decomposition of the humour-generation process could lead to various situations. It could be that all steps can be computed with resources which exist or can be computed automatically (e.g. [2]). Or some of the steps might be impossible to compute because they involve substantial unsolved problems. Here, we consider a third possible situation: where some of the steps are relatively unproblematic theoretically, but happen to be difficult or impossible to compute, for practical reasons not relevant to the simulation of humour. The solution proposed here is to use human intuition to bridge these gaps, but in a way which does not vitiate the overall model of humour. Normally human intervention in a computational model of a 'creative' process is seen as invalidating the testing of the model, but we suggest here that a suitably controlled and constrained human contribution can be wholly valid.

2 Uninformed Resource Creation

In other areas, it is routine practice to use human judgements to create data resources. Computational linguists make much use of annotated corpora, such as the Penn Treebank [3], where some quantity of language data has been marked up by human judges. Sometimes known experts are used for these tasks, but sometimes the work is done by relatively untrained persons working from explicit guidelines. Standard methodologies exist, involving (for example) checks on inter-rater agreement [4]. Often, the annotation is not specialised to some very specific subsequent use – the aim is to build a general-purpose resource, usable in whatever way other researchers see fit (not necessarily for text generation).

Similarly, in experimental psychology, it is well-established that materials to be used in an experiment may need to be pre-processed, using human reactions or judgements, to 'norm' or to 'validate' them. For example, van der Sluis and Mellish [5], in a test of the effects of 'positive' and 'negative' texts on readers, used a first stage to establish which sample texts were actually 'positive'/'negative'. Such a preliminary validation step is not part of the actual experiment, but merely calibrates the data. In such cases, the subjects may be given some explicit remit (e.g. guidelines) about how to classify the data, but quite often the calibration will be carried out by measuring reactions or effects. Such validations are relatively specialised for the particular experimental setting, even though the resource could be used elsewhere if it was suitable.

In both data annotation by specialists, and norming of data by untrained subjects, the human judge's knowledge or reactions are being used for a task which it would not be feasible to automate. However, the actual process of annotation or validation is not itself intended as a component in some wider model – it is just a practical way to create the resource. These are the direct antecedents of the methodology being proposed here.

The idea is that where a humour model requires some stage which is currently not computable automatically, the researchers should formulate an objective, explicit, modular, detailed specification of the stage, without reference to its place in the model, especially with no reference to humour. Then, using that specification, human volunteers should create, using their own judgements (and possibly existing data resources) a resource which will allow the computation of that step. This whole procedure may itself have substeps which have to be treated in the same way (a recursive application of the methodology).

In this way, the researchers can develop a step-by-step simulation of the complete humour generation process. Some of the steps will have involved human intervention and intuition – but that has occurred in isolation, 'off-line', without reference to the humour model or to humour as the purpose, and so does not undermine the claim to have a mechanistic model. The term 'uninformed' is used here to emphasise that those humans who are applying their intuition are not aware of the purpose or theoretical context of their efforts, and are not directing their intuition to the central problems of the research. In particular, steps should be taken to minimise the chance that the participants will provide (or try to provide) data which is itself 'funny' (even without the other modules in the process). In developing a theoretical account of the underlying mechanisms of humour, it is important that the overall phenomenon is decomposed into – and hence explained in terms of – components which *are not themselves humorous.* If a model of humour contained a module whose remit was 'find a funny idea', then it would be hard to argue that the model gave rise to humour – the humour would have been supplied ready-made by this single module (and the question would then be: what is inside this module to make that happen?).

3 How It Could Work

A very simple example of this approach is the way that the first prototype of the JAPE riddle generator [6] used a lexicon solicited from volunteers. However, to illustrate how a component in a processing model could be handled, consider Binsted and Ritchie's (unimplemented) model of generating 'story puns' [7]. A story pun consists of a short narrative (the setup) followed by a single-sentence (or single-phrase) punchline which is phonetically similar to some well-known phrase or saying (e.g. a proverb or quotation) and which somehow acts as a suitable conclusion to the story (often by summing up the message of the story). Binsted and Ritchie limit themselves to the case where the well-known saying is a proverb, and offer these five stages[1]:

(i) choose a maxim;
(ii) distort that maxim;
(iii) construct a semantic analysis of the distorted version;
(iv) derive constraints from that semantic form;
(v) devise a story to meet these constraints.

The first two stages should be open to automation using a collection of proverbs and some information about phonetic similarity. The third stage is in principle automatable, but difficult in the current state of the art, and the fourth stage hard to define formally. The fifth stage - story generation - is clearly challenging. To illustrate our approach, we suggest replacing stages (iii), (iv), (v) with:

(iii′) paraphrase the distorted maxim in a way that uses a different phrasing;
(iv′) given a sentence offering general advice, outline a sequence of events which would exemplify the truth of that advice.
(v′) given a sequence of events, write a narrative text describing them.

In a sense, this uses the paraphrase in place of a formal semantic representation, since this facilitates its use in guiding the story generation. This new stage (iii′) might be computable automatically, but if not, human participants could be briefed to carry out this task. Stage (iv′) would start from the results of (iii′). This could not be automated at the moment, but it could – if suitably specified – be performed by humans. That is, stage (iv′) could be computed offline by a participant who was told to list a situation, an event, or a sequence of events, which illustrated the wisdom of the advice in the supplied sentence. Similarly, (v′) could be delegated to human volunteers. It might be better to merge stages (iv′) and (v′), asking the volunteers to write short stories to illustrate the applicability of the advice, rather than having the two stages of stating an abstract spine for the story (stage (iv′)) and then having the story itself created ((v′)). Appending the distorted adage (which the stage (iv′)/(v′) volunteers have never seen) to the story would complete the construction of the story pun.

It would also be possible to have a further validation phase (for any of the stages), as in Binsted's lexicon creation, where different judges vet the data and sift out potentially faulty items.

[1] There is also an implicit sixth stage: append the result of (ii) to the result of (v).

Once again, the processing has been 'de-skilled', in the sense that non-humorous mappings (paraphrasing, story-writing) have been executed by human craft without subtler judgements about humour or the overall model.

4 Summing Up

For this approach to be viable, the humour model must have well-defined components, and any component being 'simulated' by humans must meet these criteria:

- The researcher must be able to specify clearly, objectively, and in sufficient detail what the functionality of the component is.
- The component must be modular, in order to be treated as a isolated task.
- The nature of the component must not encapsulate the essence of humour (e.g. the devising of a 'funny' text); see Section 2.
- The researcher must be able to specify the computation of that component in terms intelligible to suitable human participants who have no knowledge of the broader theoretical context.
- The specification for the participants must not reveal how the resource they create will be used – the task must have a self-contained description.

This methodological note argues that even if some component of a humour-creation model is not yet fully automatable, the model can still be tested, and a test involving human input need not be dismissed as invalid. This perspective might also be applicable in other areas of computational creativity.

Acknowledgements: Thanks to the Aberdeen NLG group for discussion of these ideas. The writing of this paper was partly supported by grant EP/E011764/1 from the UK Engineering and Physical Sciences Research Council.

References

1. Ritchie, G.: The Linguistic Analysis of Jokes. Routledge, London (2004)
2. Stock, O., Strapparava, C.: The act of creating humorous acronyms. Applied Artificial Intelligence **19**(2) (2005) 137–151
3. Marcus, M.P., Santorini, B., Marcinkiewicz, M.A.: Building a large annotated corpus of English: The Penn Treebank. Computational Linguistics **19**(2) (June 1993) 313–330
4. Carletta, J.: Assessing agreement on classification tasks: the kappa statistic. Computational Linguistics **22**(2) (1996) 249–254
5. van der Sluis, I., Mellish, C.: Towards affective natural language generation: Empirical investigations. In Mellish, C., ed.: Proceedings of the Symposium on Affective Language in Human and Machine, AISB 2008, Society for the Study of Artificial Intelligence and Simulation of Behaviour (2008) 9–16
6. Binsted, K., Ritchie, G.: Computational rules for generating punning riddles. Humor: International Journal of Humor Research **10**(1) (1997) 25–76
7. Binsted, K., Ritchie, G.: Towards a model of story puns. Humor: International Journal of Humor Research **14**(3) (2001) 275–292