Using humour to make natural language interfaces more friendly

Kim Binsted
Department of Artificial Intelligence
University of Edinburgh
80 South Bridge, Edinburgh
Scotland EH1 1HN
kimb@aisb.ed.ac.uk http://www.dai.ed.ac.uk/students/kimb

Abstract

As natural language interfaces become more widespread, much research is being carried out on ways to make them more congenial. We believe that the judicious incorporation of humorous mechanisms within the user-interface of a computer system could contribute to that goal. A limited use of humour within certain facilities — such as the signalling of errors, the reporting of the unavailability of facilities, and certain aspects of information provision — could render a computer system more user-friendly. In particular, such a system should appear less alien, less intimidating, and less patronising. The use of humour would have to be very constrained, not simply because there are limits on what is currently possible, but also because unrestricted facetiousness would be counter-productive. This could be achieved by associating the humorous behaviour with some limited part of the interface, such as a partly-anthropomorphised intelligent agent. Past success in implementing a program generating punning riddles suggests that at least some of the proposed mechanisms could be implemented in a workable way without the need for major breakthroughs.

1 Introduction

Humour is more than idle entertainment; it is a tool for aiding communication. Most people use some form of humour every day for a variety of purposes: to entertain, certainly, but also to reduce tension, to increase group bonding, to disguise ignorance and to veil criticism.

Some of the problems faced by natural language (NL) interfaces resemble obstacles which humans use humour to overcome. In particular, NL systems are often seen to be intimidating, repetitive, patronising when offering advice, and irritating when unable to respond to a request.

We speculate that computational humour might be used to make NL interfaces more friendly. Computational humour may also be of use in improving other types of interface; however, all of our work to date has been with linguistic humour, so this discussion is restricted to interfaces which can produce NL output and accept some NL input.

Certain types of humour, such as self-deprecation and humorous observation, are more appropriate for this purpose than others. Although computational humour may seem to be technically infeasible, we have already shown that simple puns can be generated by computer [Binsted and Ritchie, 1994a], and we believe that the other forms required for this purpose will not require major technical advances.

2 Past Work

Humour is used in a variety of contexts to facilitate human-human communication. In the business world, humour is considered to be important enough that a business can hire 'humour consultants' [Gibson, 1994]. Humour can be used "to criticise without alienating, to defuse tension or anxiety, to introduce new ideas, to bond teams, ease relationships and elicit cooperation" [Barsoux, 17]. Several of these goals are equally important in human-computer communication, particularly those related to reducing tension and alienation.

There has been a considerable amount of research on the linguistics of humour [Attardo, 1994; Attardo and Raskin, 1991]; however, most of the work has not been formal enough to be used directly in the computational modelling of humour, Pepicello and Green's work on puns [Pepicello and Green, 1984] being a notable exception.

Within the artificial intelligence community, most writings on humour have been speculative [Minsky, 1980; Hofstadter *et al.*, 1989]. Exceptions include DePalma and Weiner [De Palma and Weiner, 1992], who have worked on knowledge representation and riddles, Katz [Katz, 1993], who is attempting to develop a neural model of humour, and Nack and Parkes [Nack and Parkes, 1995], who are investigating humour in automated film editing.

Our work has been on the modelling and generation of puns [Binsted and Ritchie, 1994b; 1994a]. We have developed a simple model of punning riddles, which we are currently extending to cover other kinds of humour. This model is implemented in a program which generates simple punning riddles, JAPE (Joke Analysis and

Production Engine).

3 Some Problems with Natural Language Interfaces

NL interfaces have a lot of attractions. Ideally, they allow the user to make sophisticated requests without reference to a manual; they present information in an easily digestible form; and they interact with the user in a friendly, natural manner. Unfortunately, currently available NL interfaces fall somewhat short of this ideal. Most of their shortcomings are due to three main problems: limited NL coverage, limited knowledge, and 'bad manners'.

No currently available NL system can handle all possible NL input. Some can handle a variety of queries within a specific domain, while others have wider but shallower coverage. Limited coverage is often adequate for a specific application; however, there will still be queries which cannot be processed by the NL system, forcing the system to respond "I don't understand."

Even if the NL query is parsed, the system may not be able to respond appropriately because the desired information is not available. Although some systems have a great deal of knowledge on a specific subject, some queries will still go unanswered. For example, an information system about temples in Japan may have a vast knowledge base on that subject, but not be able to tell you what the weather is like right now at the main temple in Tokyo. At some point, probably quite often, a system will have to respond "I don't know."

Finally, there is the problem of the system's manners. Even occasional responses of "I don't know" and "I don't understand" can be irritating [Kaplan, 1982], and they become even more so when repeated over and over again. Error messages, designed to help the user diagnose the problem, are often perceived as patronising (e.g. "Did you remember to plug it in?"), and are sometimes ignored if repeated verbatim when the error recurs.

These issues will become less problematic as NL techniques develop and are implemented in new systems; however, none are likely to disappear soon. How important these problems are depends on the purpose of the system in question. If the system is supposed to provide information on a specific subject in response to a limited range of queries, in a context where some intimidation of the user is acceptable (e.g. a system to answer queries about train timetables), then such issues are probably not that important. If, however, a system has to respond to a wide range of queries on a number of different subjects in a friendly and polite manner (e.g. a secretarial assistant system), these problems could be critical.

4 Potential Roles for Humour

The limited use of humour within a NL interface could ameliorate some of the problems described above, making clarification queries from the NL interface less repetitive, statements of ignorance more acceptable, and error messages less patronising. These goals could be accomplished in a number of ways. We will concentrate on some of the simplest: deprecation, observational humour, and humorous referring expressions. All of these will be described in the context of a hypothetical secretarial assistant system, which should be able to respond appropriately to a wide range of commands and queries relating to the running of an office.

Self-deprecatory humour [Zillman, 1983] is often used by humans to both accept and reduce the responsibility for some sort of failure. The acceptance of the blame reduces the tension caused by the failure, while the humour reduces the importance of the failure by giving it a ridiculous cause. This kind of humour could be used by our secretarial agent to report failures, particularly when the actual cause is obvious or unimportant. For example, if the agent been instructed to search for a document, and has failed to find it, it might comment: "Sorry, I can't find any letters from S Jones. I *knew* I shouldn't have had that sixth vodka last night...".

Self-deprecation can also be used to offer advice without seeming to criticise, since it lowers the status of the one offering the advice. Error messages could be made less patronising with a little bit of self-deprecation on the agent's part: "Sorry, I don't understand the word 'parlaimentary' — perhaps I jumbled the letters on the way to the dictoinary?"

Deprecating the user or a third party is a riskier strategy, since there is a chance that the user might be offended. However, if the target is picked with care, it may be effective: "Sorry, I can't find that document. Blame the Government."

Another potentially useful genre of humour is the humorous observation. One of the advantages of observational humour is that, like self-deprecatory humour, it need not be very funny to be appreciated. If a document is taking a long time to down-load, the agent might comment: "That server is behaving like a slug in molasses. Care for a game of Tetris?"

Humorous referring expressions can be used as part of a deprecatory comment (e.g. "Sorry, I'm a bit slow today – I'm not a computer, I'm a silicon sloth...") or an observation, but they can also be selectively substituted for normal referring expressions. For example, if the user has asked the agent to contact "John", and there are several Johns to which the user might be referring, the agent might respond: "Do you want John 'not today' Bannerman or John 'beers on Friday?' Smith?" This serves the purpose of disambiguating the query, while making the agent seem more 'human'.

We suspect that the use of humour by the agent would both entertain the user, and have a more subtle psychological effect. NL interfaces, as they currently tend to be used, give the user the impression that they are talking to *the computer* (i.e. the whole system is weakly anthropomorphised). Unfortunately, this tends to exacerbate the communication problems discussed earlier, since unfriendliness and errors are then blamed on the whole system, reducing the whole system's reliability and friendliness in the user's eyes. This effect is even more marked when 'the system' includes the Internet – any failure, by any part of the system, is blamed on 'the computer'. The use of humour in an interface, however, helps to anthropomorphise a small part of the whole system (the interface 'agent'), which can then use humorous (self-) deprecation to reduce and redirect the blame for a failure. Whereas the original monolithic system was intimidating and patronising, the small anthropomorphised agent seems friendly, though fallible – an ally against the larger, impersonal system.

5 Potential Pitfalls of Humour

Although humour can ease interaction, inappropriate humour can be unbearably irritating [Adams, 1983]. If computational humour is to help make NL systems more friendly, it must be used only in the right contexts, and must be tailored to the user.

If the system (such as our hypothetical secretarial agent) is intended for a work environment, the user must be able to switch the humour off. Although humour at work is often a good idea [Barsoux, 1994], there are times when any familiar form of communication is inappropriate — and there is nothing less welcome than unwelcome humour.

If the system is to be used regularly by the same person, it is important that they be able to adjust the system's humour to their preferences. If the user cannot abide puns, for example, they should not be assaulted with punning gags ("What does a computer wear in the rain? An Apple Mac!") every time they try to print a document. Likewise, if deprecatory humour is an element of the system, the user should be able to mark certain subjects not to be targets of the machine's wit.

Some forms of humour will probably never be appropriate for the computer screen. Very aggressive or offensive humour, for example, would not suit an office environment, even if enjoyed by the user.

Finally, before humour is integrated into any NL system, the hypothesis that it would ease computer-human interaction must be tested thoroughly. We know that humour helps human-human interaction, and seems to ameliorate the kinds of problems which are faced by NL interfaces; however, we do not yet know whether computer-generated humour would have the desired effect.

6 Can It Be Done?

Humour has a lot of mystique. Linguistic humour generation is usually considered to be a strictly human skill. In science fiction, computers which are capable in every other area of human behaviour still cannot make or take a joke; those that try usually fail miserably [Adams, 1983].

In order to be successfully humorous, a NL system would need to be able to: recognise situations appropriate for humour; choose a suitable kind of humour for the situation, including a target if necessary; and generate an appropriately humorous piece of text. In section 4, we suggest some common situations (e.g. error reporting, system messages, requests for disambiguation of queries, etc) in which humour might be appropriate, and some types of humour (i.e. deprecation, observation and humorous referring expressions) which might be suitable for those situations. The remaining question, then, is whether or not a machine can generate the kinds of humour required.

Although sophisticated computational humour generation is currently out of reach, some kinds of humour can be generated using fairly simple techniques. A great deal of the simple observational and deprecatory humour described above could be produced with canned text (e.g. self-deprecatory comments attached to error messages) and simple templates. Some observational humour could be produced by modifying the NL system to use slang and exaggeration (e.g. "slow" being replaced by "as slow as a slug in molasses") when humour was needed. If humour is to be used flexibly and intelligently, however, we need to develop and implement models of the kinds of humour required.

JAPE [Binsted and Ritchie, 1994a] is an implementation of a simple model of puns, which is being gradually extended to cover a wider range of humour. The original model, implemented in JAPE-1, describes question-answer punning riddles with noun-phrase punchlines. By substituting homonyms into common noun phrases, JAPE-1 constructs nonsense phrases and their meanings, then asks questions about these constructions. The questions and their answers, being about nonsense objects, are riddles. For example, the riddle:

> What kind of pig can you ignore at a party? *A wild bore.*

is made up of a question and answer about the constructed lexical item:

> **wild bore:** a type of pig which is boring.

However, the schemata which JAPE-1 uses can also describe more complex constructions than substitutions into noun phrases. For example, one new schema does not begin with a genuine noun phrase at all, but instead constructs noun phrases entirely from homophones, generating nonsensical lexical items such as:

> **bizarre bazaar:** a weird marketplace.

Schemata of this more general type, which do not rely on substitution into common noun phrases, describe a large variety of puns. Moreover, because the schemata construct lexical items which are then used as if they were genuine (i.e. the generation of the question-answer form is not humour-specific), the way in which the pun is used can be as sophisticated as the NL abilities of the system allow. For example, the same pun which is used in the riddle:

> How do you make gold soup? *Put fourteen carats in it!*

could be used by our hypothetical secretarial agent to cover a gap in its knowledge:

> Q: How many carats is the jewelry in the next shipment?
> A: Why, were you planning to make a salad out of it?? Sorry, I don't know...

JAPE is currently being modified to be part of a NL interface to a MOO (a NL virtual environment) [Loehr, 1995]. The MOO is intended to be an environment for students to learn English by interacting linguistically with the environment and with each other. The NL system being constructed will reside in a 'robot' guide which will be able to converse with the students, answer simple NL queries, and (if the integration of JAPE is successful) make simple, relevant puns.

Although the ability to pun is not sufficient to generate all the kinds of humour described in section 4, we believe that our success in modelling and generating puns suggests that the computer generation of other simple forms of humour is feasible.

7 Conclusion

Humour is used by humans to ease all sorts of communications problems. Some of these problems — patronising or intimidating advice, irritating ignorance and repetitive apologies — also arise in human-computer communication, and we propose that computational humour could be used to solve them. The kinds of humour required for this purpose are not very sophisticated, and could be generated by a computer without any major technical advancements. Used sparingly and carefully, humour could make NL interfaces much friendlier.

Acknowledgements

This work is financially supported by NSERC Canada and the Overseas Research Studentship Scheme. My supervisors are Graeme Ritchie and Helen Pain.

References

[Adams, 1983] Douglas Adams. *The Hitchhiker's Guide to the Galaxy.* 1983.

[Attardo and Raskin, 1991] Salvatore Attardo and Victor Raskin. Script theory revis(it)ed: joke similarity and joke representation model. *Humor*, 4(3):293–347, 1991.

[Attardo, 1994] Salvatore Attardo. *Linguistic Theories of Humour.* Mouton de Gruyter, Berlin, 1994.

[Barsoux, 17] Jean-Louis Barsoux. The entertainment industry: the peculiar use of humour in Britain's businesses. *Times Higher Education Supplement*, page 22, March 17.

[Barsoux, 1994] Jean-Louis Barsoux. *Funny Business.* Cassell, 1994.

[Binsted and Ritchie, 1994a] Kim Binsted and Graeme Ritchie. An implemented model of punning riddles. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, Seattle, USA, 1994. Available as Research Paper 690, Department of Artificial Intelligence, University of Edinburgh.

[Binsted and Ritchie, 1994b] Kim Binsted and Graeme Ritchie. A symbolic description of punning riddles and its computer implementation. *submitted to Humor*, 1994.

[De Palma and Weiner, 1992] Paul De Palma and E. Judith Weiner. Riddles: accessibility and knowledge representation. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-92)*, volume 4, pages 1121–1125, 1992.

[Gibson, 1994] Donald E. Gibson. Humor consulting: laughs for power and profits in organizations. *Humor*, 7(4):403–428, 1994.

[Hofstadter *et al.*, 1989] D. Hofstadter, Liane Gabora, Victor Raskin, and Salvatore Attardo. Synopsis of the workshop on humor and cognition. *Humor*, 2-4, 1989.

[Kaplan, 1982] S. J. Kaplan. Cooperative responses from a portable natural language data base query system. *Artificial Intelligence*, 19:165–187, 1982.

[Katz, 1993] Bruce F. Katz. A neural resolution of the incongruity-resolution and incongruity theories of humour. *Connection Science*, 5(1):59–75, 1993.

[Loehr, 1995] Daniel Loehr. Using JAPE for a NL MOO robot. personal communication, March 1995.

[Maes, 1991] Patti Maes, editor. *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back.* Bradford Books/ MIT Press, 1991.

[Minsky, 1980] Marvin Minsky. Jokes and the logic of the cognitive unconscious. Technical report, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1980.

[Nack and Parkes, 1995] Frank Nack and Alan P Parkes. Automated film editing for educational applications? don't make me laugh! A paper proposed for ED-MEDIA '95, 1995.

[Pepicello and Green, 1984] W.J. Pepicello and Thomas A. Green. *The Language of Riddles.* Ohio State University, 1984.

[Zillman, 1983] D. Zillman. Disparagement humour. In P. E. McGhee and J. H. Goldstein, editors, *Handbook of Humour Research*, pages 85–108. Springer Verlag, 1983.