A Measure of Funniness, Applied to Finding Funny Things in WordNet

Jonas Sjöbergh
Hokkaido University
js@media.eng.hokudai.ac.jp

Kenji Araki
Hokkaido University
araki@media.eng.hokudai.ac.jp

Abstract

We generate two new types of jokes based on ambiguity and spelling similarity in multi-word expressions in WordNet. Since many similar jokes can be generated for each multi-word expression, we also use a measure of the funniness of words to select which of the possible joke candidates is likely to be the funniest. Evaluation shows that the funniness measure does on average find funnier candidates than the baseline, which in turn outperforms selecting the least funny candidate. The funniness measure thus ranks candidates according to how funny they are. The differences between the methods are quite small though, so the ranking is less than perfect. Comparing the generated jokes to human made jokes shows that the system is not as funny as professional human made humor, but the better jokes achieve human level.

1 Introduction

Computational humor has not seen that much research done but a good overview of the work so far is given in (Binsted et al., 2006). The research can be divided into humor recognition (Taylor and Mazlack, 2005; Mihalcea and Strapparava, 2005; Sjöbergh and Araki, 2007a) and humor generation (Binsted, 1996; Binsted and Takizawa, 1998; Tomoko et al., 1996; Yokogawa, 2001; Stark et al., 2005; Sjöbergh and Araki, 2007b, 2008)

In this paper we deal with humor generation. It is common for humor generation systems to generate many not very funny jokes and a few very funny jokes. Some way of measuring which jokes are funny and which are boring would help to improve the quality of humor generation systems. We here present two new methods of generating jokes in English based on compound nouns in WordNet. Both methods use many types of information from WordNet to generate jokes that are sometimes quite funny.

They do however suffer from the problem that for a single compound noun sometimes tens of jokes can be generated, all of them almost the same joke so only one candidate should be selected. Among these candidates many are boring or hard to understand while some are significantly more funny than the others. We thus also present a method for measuring the funniness of jokes, so as to automatically select which candidate is the best when there are many potential jokes.

Our measure of funniness is a rather shallow measure, no deep analysis of the jokes are done. Basically, if there are many words that are correlated with jokes (i.e. occur more often in jokes than in “normal text”) then the candidate is believed to be funnier than if there are few words correlated with jokes.

2 Measuring Funniness of Words

For computers it is difficult to understand what is funny and what is not funny. For humor generation systems, such understanding would be helpful though, automatically removing boring jokes would increase the quality a lot. What is perceived as funny is of course very subjective and thus difficult to formalize. Humor is also very complex, many things have an influence on what is funny. Having a deep understanding of why something is funny is liekly too difficult for current AI or language processing systems, but simple measurements can still be of use.

We use a simple measurement to measure how funny a certain word is, which is based on the idea that words that occur often in jokes are likely to be funny while words that rarely occur in jokes are likely to not be very funny. If a word is funny it should be easier to use in jokes (where the purpose is to be funny) than if it is not funny, and occurrence in jokes should thus be correlated with funniness of words. Especially in our application, where we want to filter out funny joke candidates from unfunny ones, high occurrence of words common in jokes is likely to correlate with funniness even if it is not true that the words in themselves are funny.

To select if “coin bank” being similar to “corn bank” is funnier than being similar to “join bank” we thus use a simple measure for the funniness of each of the words. We have collected a corpus of almost 7,000 oneliner jokes. For comparison we created a corpus of the same number of randomly selected sentences from the British National Corpus, BNC (Burnard, 1995), of lengths similar to each joke. We also downloaded 2,300 jokes from the top list of the joke news group rec.humor.funny[1]. For comparison we extracted one paragraph with roughly the same number of words from the BNC for each joke. The funniness of a word is based on whether the word is more common in jokes than in “normal text” (here meaning the BNC).

The funniness of a word that occurs both in the joke corpus and in the non-joke corpus is simply the word frequency in the joke corpus divided by the frequency in the non-joke corpus. Other statistical measurements could of course also be used, for instance based on Bayes rule or χ^2 measurements. If the word only occurs in the joke corpus the funniness is twice the frequency in the joke corpus. If the word occurs only in the non-joke corpus, the frequency in the joke corpus is counted as 0.1, and funniness is calculated as before. If the word does not occur in either corpus the funniness is 0. This means that words that occur only in the non-joke corpus are treated as funnier than words that do not occur in either corpus, which is intentional. Words that do not occur in either corpus are normally rare and difficult words (common in WordNet) that most readers would not understand, making any joke with them unfunny. Occurrence in the non-joke corpus at least shows that the word is somewhat common, and thus slightly more likely to be funny.

There is also a short stop list of words that are not funny but frequent, such as “do”. Words that are inappropriate can also be added to the stoplist to avoid jokes on topics that are hard to laugh about. Stoplist words get a funniness of 0.

[1]http://www.netfunny.com/rhf/

Since dirty jokes are popular, we also rank up jokes with dirty words in them. We have a collection of 2,500 dirty words and expressions in English. Only 1,200 are single words, the rest are multi-word expressions and are currently not used in our funniness measure. If a word is listed as a dirty word, the funniness of the word is multiplied by 10. Most of the jokes that are generated by the system have no candidate that contains words that are recognized as dirty words, so this weighting does not have that much of an impact.

Finally, since WordNet contains a lot of Latin names for plant and animal species, and these are normally not funny because people do not know what they mean, we also divide the funniness of any capitalized word by five. This also penalizes names of people, which we think is OK since we are not trying to make jokes about specific people.

3 Generating WordNet Jokes

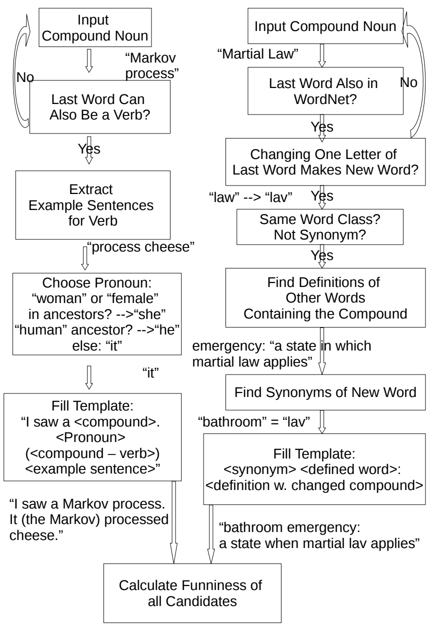
We generate two new types of jokes based on possible ambiguity or similarity in compound nouns in WordNet. The methods use several types of knowledge included in WordNet, such as usage examples, word class information, whether something is a living thing or not, if it is male or female, etc. The generated joke types are by no means extremely funny, but the methods can generate jokes that are entertaining. They can however generate very very many jokes, and many of them are not very funny at all. To find the funniest jokes, we use the funniness measure from the previous section.

An overview of the generation procedures is given in Figure 1. These types of jokes turn out to be fairly easy to generate for a large number of words, and can thus be used when there is a need for jokes related to some specific word, such as when a chat-bot wants to make a joke related to something the user has said.

3.1 Compound Nouns With Verbs

In our joke corpus there are jokes on the form “If you saw a heat wave, would you wave back?”, i.e. using the fact that “heat wave” ends with the word “wave” which can also be interpreted as a verb.

We generate such jokes based on WordNet compound nouns. For a compound, the last word in the compound must also occur as a verb in WordNet and the rest of the compound must occur as a noun. For example “fish stick” is accepted since “fish” is a noun and “stick” can also be a verb.



Input Compound Noun
"Markov process"
No
Last Word Can Also Be a Verb?
Yes
Extract Example Sentences for Verb
"process cheese"
Choose Pronoun: "woman" or "female" in ancestors? -->"she" "human" ancestor? -->"he" else: "it"
"it"
Fill Template: "I saw a <compound>. <Pronoun> (<compound – verb>) <example sentence>"
"I saw a Markov process. It (the Markov) processed cheese."
Input Compound Noun
"Martial Law"
Last Word Also in WordNet?
No
Yes
Changing One Letter of Last Word Makes New Word?
"law" --> "lav"
Yes
Same Word Class? Not Synonym?
Yes
Find Definitions of Other Words Containing the Compound
emergency: "a state in which martial law applies"
Find Synonyms of New Word
"bathroom" = "lav"
Fill Template: <synonym> <defined word>: <definition w. changed compound>
"bathroom emergency: a state when martial lav applies"
Calculate Funniness of all Candidates

Figure 1: Flowchart for generating jokes.

WordNet example sentences and definitions are then extracted for this verb. For the example verb "stick", there is an example sentence "stick your thumb in the crack".

To find an appropriate pronoun to use for the noun when generating a sentence pointing out the other possible interpretation the WordNet hierarchy is used. Since the subject for the verb is the remainder of the compound noun when removing the verb, a pronoun for this noun is needed. If the noun has an ancestor in any number of steps in WordNet that is either "woman" or "female", the pronoun used will be "she". If the noun is not related to "woman" or "female" but is related to "human" then the pronoun "he" will be used. Otherwise "it" will be used. For the example word, the remaining words when removing "stick" is only "fish", and "fish" has "female" as one of its ancestors so the pronoun "she" is selected.

The output is currently based on a simple template, by starting with "I saw" and then adding the compound noun. This is then followed by the example sentence, with the original subject replaced with the appropriate pronoun (also replacing any genitive pronouns etc. in the rest of the example sentence). To make it extra clear what interpretation is intended the pronoun is followed by an explanation of what it refers to in parenthesis, i.e. the noun. So for "fish stick" one possible output is "I saw a fish stick. She (the fish) stuck her thumb in the crack." Since there are often many example sentences for a verb, several jokes can be generated. Sentences with this verb found in other places than WordNet could of course also be used.

3.2 Changing One Letter in Compound Nouns

Another type of joke in our joke corpus is jokes like "Acupuncture: a jab well done" or "Shotgun wedding: a case of wife or death", i.e. making use of the facts that "life or death" is fixed expression and that "wife" and "life" are very similar.

To generate simple versions of jokes like these, we make use of WordNet compound nouns again. The compound must contain a word that is also included in WordNet by itself and that by changing one letter in the word becomes another word in WordNet. There is of course no need to restrict the change to be just one letter, one could instead use for instance sound similarity or some other similarity measure, though we restrict ourselves to spelling differences of one letter for now. The new word must be an adjective if the original word can be an adjective, and must be a noun if the original word is a noun but cannot be interpreted as an adjective. For example, the compound noun "god of war" can be changed into "god of car" by changing one letter of "war". Both "war" and the resulting "car" are nouns in WordNet.

The new word must not be a synonym of the original word and the new compound noun not a synonym of the original compound. This is checked using the synonymy information in WordNet. This avoids generating "grey cat" from "gray cat" and other similar spelling variations.

When a candidate compound noun has been found, definitions for nouns in WordNet containing this compound are extracted (usually not definitions of this compound noun). All words that have the compound in one of their definitions are extracted. In our example of "god of war", nouns like "Ares" and "Mars" are extracted together with their definitions "(Greek mythology) Greek god of war" and "(Roman mythology) Roman god of war" respectively.

Synonyms of the new word created by changing one letter of one of the original words are also extracted. In our example, "auto", "automobile", "machine", and "gondola" are extracted as synonyms of "car". The joke is then presented by using one of these synonyms as a modifier for one of the words with definitions. The definition itself is then also presented, but with the original compound replaced by the new compound. So in our example, one possible output joke is "auto Ares: (Greek mythology) Greek god of car".

3.3 Filtering Forth Funny Jokes

Both methods detailed above can in general generate very many jokes for each compound noun. Most of these will be very similar, so seeing one is enough and seeing more will just be boring. Many of the possible jokes are not very funny, for instance because they use words that a normal reader would not know (WordNet contains many obscure words) or words that are not very funny, while other jokes can be very funny. To filter out which possible joke for a compound noun is the funniest and then discard the other possibilities we use the previously mentioned funniness measure.

A simple measure of the funniness of a joke is to calculate the average funniness of the words in the resulting output. This works to some extent, but since some words are more important than others in the jokes, it is a little bit too simplistic. It is for instance more important that the newly created compound in the spelling similarity method contains funny words than that the rest of the definition contains funny words. The synonym and the word defined used to introduce the new compound are also quite important.

For these jokes, the funniness value of the joke is calculated as three times the funniness of the new word plus two times the funniness of the synonym used to describe the new word plus the funniness of the total generated text. For the jokes using the noun/verb ambiguity the funniness of the joke is simply the funniness of the total generated text since all jokes have the same structure except for the example sentence used and thus measuring only the total text gives good results.

4 Evaluation

To evaluate the funniness measurement, we generated jokes of the previously explained types. First we asked evaluators to rank the funniest (according to the system) candidate and the least funny candidate based on the same compound. We also added a baseline, which always selects the shortest (in number of characters) candidate. Long sentences tend to be complex, be more likely to contain unknown or difficult words, and have more content that distracts from the punch line. Thus, the shortest possible joke is a reasonable baseline.

For each compound that the system can generate at least two candidates with different funniness measurements for, the evaluators thus had to rank three suggestions, the funny candidate, the boring candidate, and the baseline candidate. The candidate that the evaluator thought was the funniest version of a joke based on this compound was assigned rank 1, the second funniest candidate rank 2, etc. Sometimes the baseline selected the same candidate as one of the other methods, in which case these two were ranked the same. The evaluators were allowed to rank several candidates as equally funny if they could not decide which one they liked better.

The candidate jokes were ordered alphabetically, so which method's candidate would appear in what position when shown to the evaluator did not depend on the method, and all methods occurred at all positions. The evaluators ranked candidates for jokes based on ten compounds for each joke generation method, i.e. ten compounds for the noun/verb ambiguity method (giving 30 such candidates), and ten compounds for the spelling similarity method (another 30 candidates).

The evaluators also rated a few other jokes on a scale from 1 (boring) to 5 (funny). These jokes were also generated by the two WordNet based methods but only the candidate believed by the system to be the funniest was used. We also included three human made jokes of the same types. These jokes were taken from our joke corpus and used as a reference to see how funny the system is compared to humans and how funny these types of jokes are. Since there are not many jokes of these types in our corpus, we only found one joke of the noun/verb ambiguity type and two jokes of the spelling similarity type. Five system generated noun/verb ambiguity jokes and six system generated spelling similarity jokes were used in this part of the evaluation. All evaluators evaluated the same set of jokes. The reasons for not including more jokes in the evaluation are that we had difficulties in finding evaluators fluent in English, and giving each evaluator more work than we did would make most fluent speakers that we did find decline to participate. Nine evaluators, all fluent speakers of English though not all were native speakers, took part in the evaluation.

Candidate	Spell. Similarity	Noun/Verb
Least Funny	2.41	2.12
Funniest	1.71	1.91
Baseline	2.07	1.97

Table 1: The average ranking (1=funniest, 3=least funny) of the suggestions from the different candidate selection methods.

For both parts of the evaluation, which compounds to use were selected randomly. This means that there could be funnier jokes than the ones that were selected for the evaluation.

The results of the ranking of candidates is shown in Table 1. It can be seen that selecting the candidate that the system believes is the funniest candidate does indeed give the best ranking. In the same way, selecting the least promising candidate gives a worse average ranking than the baseline of always selecting the shortest possible output. The ranking does thus do what it is supposed to do. The difference is however not that big, so the ranking is not as good as could be hoped for. Especially for the noun/verb ambiguity jokes the differences between the methods are very small, which is in large part due to the fact that there are usually very few candidates for these jokes, so the baseline method almost always selects the same candidate as one of the other methods effectively reducing the number of positions in the rankings. Many of the candidates are also very similar in funniness even when slightly different, with differences being things such as “ditch a car” compared to “ditch a plane”.

For the spelling similarity jokes there are generally many more candidates available, thus usually giving a larger difference between the best and the worst candidate, and the differences are also much more important for the joke. All noun/verb ambiguity jokes have the same basic idea, the same noun is reinterpreted as the same verb in each candidate. In the spelling similarity jokes the different candidates build on changes of one word to a different word, and the new word is different in the different candidates. The difference between the best candidate and the baseline candidate is significant (Students t-test, 5% level) for the spelling similarity jokes, but the difference for the noun/verb ambiguity jokes is not significant.

That the system does not achieve a perfect ranking is not surprising, since judging the funniness of a joke based on only the funniness (or frequency of occurrences in other jokes) of the words in the joke is very simplistic. The correlation between occurrences of funny words (words common in jokes) and the candidate being funny does seem to be important though, since the best candidate is ranked better than both the baseline and the worst candidate, and the worst candidate is (as it should be) ranked worse than the baseline.

Type	Man Made	System
Spelling Similarity	4.3	2.6
Noun/Verb Ambiguity	3.4	2.6

Table 2: The funniness of human made jokes and a random sample of system generated jokes.

The agreement between the evaluators varied between jokes, on average 5.5 evaluators chose the same ranking for the spelling similarity jokes and 5.2 for the noun/verb ambiguity jokes. These figures are a little lower than the actual agreement, since different evaluators scored ties in different ways (e.g. place 1 for the top candidate and place 2 for the other two or place 1 for the top candidate but place 3 for the other two). While the agreement is lower than could be hoped, no agreement at all between would give only slightly over 3 evaluators choosing the same ranking.

The funniness of the top ranked candidates and of the human made jokes are shown in Table 2. The human made jokes are as expected funnier than the system made jokes. Humor is difficult and many factors influence whether a joke is funny or not, so even state of the art humor generation systems generally fare very badly compared to humans. The scores are still quite promising, and the highest scoring system made jokes in the evaluation all scored 3.4, which is the same as the lowest ranked human made joke.

The agreement between the evaluators was fairly high, with on average four evaluators assigning the same score to the computer generated jokes and on average five evaluators for the human made jokes (no agreement at all would give slightly less than two evaluators agreeing). Since the exact level of funniness is so subjective, this is quite high. One human made joke for instance had five evaluators voting “5” and the other four voting “4”, so while the number of evaluators agreeing is only five, all agree that it is funny joke.

In the future we plan to normalize the funniness measure so it can be used to compare different jokes to each other too, so that not only the funniest candidate of all jokes based on the same compound can be selected but the funniest joke of all the jokes that can be made from WordNet can be found automatically.

5 Conclusions

We presented two new methods for automatically generating jokes based on compound nouns. Both of these have the problem that for each compound there can be very many possible ways to generate a joke, but all of these are very similar so only one should be selected and used. We presented a measurement to compare the funniness of such candidates, to be used to select the funniest candidate automatically. This method is based on measuring the funniness of words based on how often they occur in jokes in a large joke database compared to how often they occur in non-joke texts.

The evaluation showed that the candidate that is funniest according to the system is ranked as funnier on average than the worst candidate and also funnier than the baseline method of just selecting the shortest possible joke candidate. Thus, the system does find funnier candidates by using the measurement of funniness. Conversely it also correctly disregards unfunny candidates, the least funny candidate was ranked lower than the baseline method. The system is of course not correct in every case, though.

When comparing the funniness of system generated jokes to man made jokes, the system is unsurprisingly still not as funny as professional joke makers. The best scoring jokes achieved human level, though.

References

Kim Binsted. 1996. *Machine Humour: An Implemented Model of Puns*. Ph.D. thesis, University of Edinburgh, Edinburgh, United Kingdom.

Kim Binsted, Benjamin Bergen, Seana Coulson, Anton Nijholt, Oliviero Stock, Carlo Strapparava, Graeme Ritchie, Ruli Manurung, Helen Pain, Annalu Waller, and Dave O'Mara. 2006. Computational humor. *IEEE Intelligent Systems*, 21(2):59–69.

Kim Binsted and Osamu Takizawa. 1998. BOKE: A Japanese punning riddle generator. *Journal of the Japanese Society for Artificial Intelligence*, 13(6):920–927.

Lou Burnard. 1995. The Users Reference Guide for the British National Corpus.

Rada Mihalcea and Carlo Strapparava. 2005. Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of HLT/EMNLP*. Vancouver, Canada.

Jonas Sjöbergh and Kenji Araki. 2007a. Recognizing humor without recognizing meaning. In Francesco Masulli, Sushmita Mitra, and Gabriella Pasi, editors, *Proceedings of WILF 2007*, volume 4578 of *Lecture Notes in Computer Science*, pages 469–476. Springer, Camogli, Italy. URL `http://dr-hato.se/research/clip2007.pdf`.

Jonas Sjöbergh and Kenji Araki. 2007b. Recreating humorous split compound errors in Swedish by using grammaticality. In *Proceedings of Nodalida 2007*, pages 389–393. Tartu, Estonia. URL `http://dr-hato.se/research/avigt07.pdf`.

Jonas Sjöbergh and Kenji Araki. 2008. A complete and modestly funny system for generating and performing Japanese stand-up comedy. In *Coling 2008: Companion volume: Posters and Demonstrations*, pages 109–112. Manchester, UK.

Jeff Stark, Kim Binsted, and Benjamin Bergen. 2005. Disjunctor selection for one-line jokes. In *Proceedings of INTETAIN 2005*, pages 174–182. Madonna di Campiglio, Italy.

Julia Taylor and Lawrence Mazlack. 2005. Toward computational recognition of humorous intent. In *Proceedings of Cognitive Science Conference 2005 (CogSci 2005)*, pages 2166–2171. Stresa, Italy.

Kanasugi Tomoko, Matsuzawa Kazumitsu, and Kasahara Kaname. 1996. Applications of ABOUT reasoning to solving wordplays [in Japanese]. *IEICE technical report. Natural language understanding and models of communication*, 96(294):1–7.

Toshihiko Yokogawa. 2001. Generation of Japanese puns based on similarity of articulation. In *Proceedings of IFSA/NAFIPS 2001*. Vancouver, Canada.