Puntective - A System for Pun and Wordplay Detection

Cody Hanson
University of Colorado Colorado Springs
1420 Austin Bluffs Pkwy
Colorado Springs, Colorado USA 80918
chanson@uccs.edu

Abstract

Puns and wordplay are easy for humans to comprehend, but difficult for computer programs. With the right algorithm, perhaps computers could mimic the way that humans can make connections between words, sounds, the current social setting, all while drawing from deep knowledge bases. This research proposes a set of heuristics and techniques for allowing a computer to assign a 'pun confidence score' to a piece of text.

Introduction

Written language is one of the crowning achievements of humans, and one of our most sophisticated and complex tools. An important part of this medium is that of humor and jokes. Humorous writing works to bring special joy to humans, and we are able to craft nuanced and smart jabs that require much thought to appreciate. What would it take to teach a computer the concept of 'humor'? How about just identifying its presence in written or spoken word? An especially complex form of humor is the 'Pun'. A pun is often referred to as a 'play on words' , and while easy for most humans to recognize, present a challenge to computers and the field of natural language processing,

In this paper we propose a system of heuristics and processing that attempts to recognize whether a piece of text constitutes a pun. By using techniques of natural language processing and heuristics for finding common properties of puns we will assign a 'confidence score' which will attempt to quantify the likelihood that the piece of text is a pun. Equally interesting is the estimation by a computer that a piece of text *doesn't* have any humorous content. Ideally a good algorithm will be designed to handle both of these cases.

Related Work

(Stamatatos, Fakotakis, and Kokkinakis 2000) discusses a technique for classifying the 'type' of publication that some text is from, depending on the word content and frequency. This approach was done with discriminant analysis, and a 'training' dataset. It was not limited to humorous text.

In (Kao, Levy, and Goodman), a computational model for humor is presented, dealing specifically with homophone puns. The authors limited themselves to this subset in order to narrow the problem scope. This work shows that it is possible to 'quantify' something like word play, and should serve as a great knowledge base for my own research. These authors used the 'Bag of Words' approach when doing their analysis.

(Ritchie 2005) presents a method for pun generation using computers, as well as examines the 'essence' and structure of different kinds of puns.

Proposed Methodology and Heuristics

The first step will be to read in the words of the text under examination. The first naive approach will to just count the frequency of each word in a 'bag of words' approach. This will be simple to implement initially, and won't require deep knowledge of natural language processing (NLP) algorithms that would be required to parse the sentence into a tree-like structure. Once the bag of words is constructed we can apply several proposed heuristics.

If the word frequency counting proves to not be powerful enough to capture the essence of the text, NLP algorithms such as those demonstrated by (Socher et al. 2013) will be researched in order to augment the approach.

In order to maximize development speed, and focus on actually solving the problems, we will use simple console based programs that leverage a scripting language such as Python. If needed, we will organize our backing data into a SQLite database, to minimize complexity and enable easy sharing with other academics. SQLite will also provide for a powerful querying mechanism while implementing code.

Odd Hyphenations

A characteristic of puns that the authors have noticed is that they can sometimes contain 'odd' hyphenations, or more precisely, hyphenations of words that are not commonly seen.

Figure 1: Puns with Odd Hyphenations
Atheism is a non-prophet organization. A bicycle can't stand on its own because it is two-tired.

In Figure 1, 'non-prophet' (rather than 'non-profit') and 'two-tired' (rather than 'too-tired') are hyphenations which most humans would deem as atypical. My idea would be to look for hyphenated phrases in a dictionary, or other datastore of figures of speech, and if the phrase isn't found, then we can perhaps make the assumption that it is a form of wordplay.

Homophones

Puns often of homophones in them. A homophone is defined as being each of two or more words having the same pronunciation but different meanings, origins, or spelling, e.g., new and knew

Figure 2: Pun with Homophones
What did the grape say when it got stepped on? Nothing - but it let out a little whine.

In Figure 2, the wordplay using homophones is 'whine' which is a play on 'wine' which is related to 'grape'. By looking for homophones of 'whine', we would be led to 'wine', which we could then make a linkage to grapes and winemaking, using a dictionary or encyclopedia.

Dictionary Definitions

Figure 3: Pun with terms linked by dictionary definitions
I used to be addicted to soap, but I'm clean now.

In Figure 3, we can see some key words, 'addicted', 'clean' and 'soap' have the connections between them that constitute a pun. For this example, we would process 'addicted', and possibly find references to 'clean', and the dictionary definition of 'soap' could also have references to 'clean'. When we see that 'clean' has different definitions, with rather disparate meaning, we may be able to glean that there is a pun present.

One letter mutations

Figure 4: Pun with relevant 1 letter mutation
Did you see the movie about the hot dog? It was an Oscar Wiener.

In Figure 4 it can be seen that 'Wiener' is one letter removed from being the word 'Winner', which is an appropriate reference to movies and the Oscar's awards show. By looking for other words which have a single letter different, it can help us to make linkages to the other parts of the sentence, using our other techniques.

Synthesized speech with dictionary search

In Figure 5, the word play of the phrase comes out when it is spoken. In this case, the Knight's name 'Sir Cumference' becomes a homophone with 'circumference'. A stretch goal for this research could be to process text with text to speech

Figure 5: Pun with audible word play
The roundest knight at king Arthur's round table was Sir Cumference.

software, and then back again from the generated audio, back into text. If the transformed text differs from the input text, it could be indicative of audible word play. Of course this technique has its difficulties, and depends upon the quality of the text to speech, and speech to text synthesizers.

Evaluation

In order to judge the effectiveness of the system, we will compare the results of the pun classification to that of a human expert. We will build a dataset that contains a mixture of text snippets that contain puns, and some that do not. Also, because not all puns are created equal, we will seek out puns of varying complexity (as judged by the number of 'connections' a human can make) to attempt to highlight limitations of our techniques. To ensure that the evaluation is fair, the texts in the dataset will be selected with as much randomness as possible, from a pool of candidates, and the algorithms will not be tailored in any way to a specific piece of text.

It will be important to track statistics on success rates, including false positives. False positives will indicate that the techniques are not refined enough, or are going down a path of incorrect analysis.

Speed of execution will not be a main goal or metric for this research. It is hopeful that search techniques will take at most on the order of a few minutes, but if the complexity of the analysis become of too high, those techniques will need to be rethought.

Conclusion

This research topic takes on a difficult problem, teaching computers about humor. Due to the limited time available for the work, the authors will strive to make meaningful progress for a manageable subset of the problem.

References

Kao, J. T.; Levy, R.; and Goodman, N. D. The funny thing about incongruity: A computational model of humor in puns.

Ritchie, G. 2005. Computational mechanisms for pun generation. In *Proceedings of the 10th European Natural Language Generation Workshop*, 125–132.

Socher, R.; Bauer, J.; Manning, C. D.; and Ng, A. Y. 2013. Parsing With Compositional Vector Grammars. In *ACL*.

Stamatatos, E.; Fakotakis, N.; and Kokkinakis, G. 2000. Text genre detection using common word frequencies. In *Proceedings of the 18th Conference on Computational Linguistics - Volume 2*, COLING '00, 808–814. Stroudsburg, PA, USA: Association for Computational Linguistics.