Identification of Homographic Pun Location for Pun Understanding

Yu-Hsiang Huang Hen-Hsen Huang Hsin-Hsi Chen
Department of Computer Science and Information Engineering, National Taiwan University
No. 1, Sec. 4, Roosevelt Road, Taipei 10617, Taiwan
{yhhuang, hhhuang}@nlg.csie.ntu.edu.tw, hhchen@ntu.edu.tw

ABSTRACT

This paper introduces a novel framework for homographic pun location identification. Two observations, the existence of the support term and the preferred positions of the pun in context, are considered as crucial hints for pun identification. We first nominate the pun candidates, and then select the most probable one based on various strategies. Experimental results show the effectiveness of our method.

Categories and Subject Descriptors

Computing methodologies → Lexical semantics

Keywords

Pun understanding; Homographic; Computational humor

1. INTRODUCTION

Pun, also called paronomasia, is a subtle rhetorical device which derives the humor by creating a surprise based on the ambiguity of two or more interpretations. Puns are widely used in our daily life like advertising slogans because the brief and delicate wordplay makes them memorable. The ambiguity in puns can be roughly classified into four types: meaning level, interpretation level, grammar level, and phonological level.

(S1) plays an ambiguity at the meaning level. The pun is the noun *interest*, which has two meanings: "a sense of concern or curiosity" and "a fixed charge for borrowing money." In (S2), the pun *reflection* is actually in the same word meaning, but different interpretations can be made, i.e., a reflection of light or personality. (S3) shows a pun at the grammar level. The word *leaves* can be a noun (the plural of *leaf*) or a verb (the third person singular of *leave*). (S1), (S2), and (S3) are the instances of the homographic pun, in which different meanings share the same spelling [3]. (S4) shows a homophonic pun, where the pun is at the phonological level. The word *pore* sounds similar to the other word *poor*.

(**S1**) I used to be a banker, but I lost **interest**.

(**S2**) It's a clumsy **reflection** of yourself when you break a mirror.

(**S3**) When the nomadic tree senses danger it packs up its trunk and **leaves**.

(**S4**) If you say you have bad skin, I'd say that was a **pore** excuse.

In pun understanding, previous works focus on the interpreation of a given pun. Miller and Gurevych [3] proposed a word sense disambiguation approach to suggest the two senses of a homographic pun. Jaech et al. [2] described a weighted finite-state transducer to recover the homophonic pun. Both works assume the location of the pun in the context is known. Compared to pun interpretation, identifying the pun from its context is challenging and not yet addressed. Based on our observations on puns, we propose a novel framework to locate the homographic pun in punny jokes. The effectiveness of our approach is discussed.

WWW'17 Companion, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3054257

2. TWO OBSERVATIONS ON PUNS

From the analysis on punny jokes, two observations are given to identify the location of a pun: 1) the existence of a support term that hints a secondary meaning of the pun, and 2) the pun is likely to locate in the rear part of its context.

2.1 The Existence of the Support Term

On the one hand, the context of a pun will suggest readers a common interpretation. On the other hand, there exists a support term which has a strong correlation with another sense of the pun. The support term gives readers a hint about the second interpretation. In (S1), for example, *banker* is the support term for the pun *interest*. The first meaning of *interest*, a sense of concern or curiosity, can be understood even if the word *banker* is replaced with another job title like *engineer*. In contrast, the second meaning of *interest*, a fixed charge for borrowing money, is unrevealed without the hint from *banker*. In other words, a pun cannot stand without a support term. Thus, the support term is important to pun identification.

2.2 Preference Positions of the Pun in Context

To our observation, the pun is more likely to appear in the rear part of its context. Like the punchline in a joke, the pun will have a more powerful effect if there is a delicate elaboration beforehand. In (S1), (S3), and (S4), the puns stand in the second half part. We calculate the position ratio of the puns in the dataset of Jaech et al. [2] by dividing the pun location index by the length of the whole instance. The mean position ratio is 0.73 with a standard deviation of 0.25. This statistics supports our observation.

3. A FRAMEWORK FOR PUN LOCATION

Our framework for pun location identification consists of two stages. The first stage suggests the pun candidates given an instance. The score of each candidate is also computed. The second stage selects the most probable one from the candidates to obtain the final result.

3.1 Nomination of the Pun Candidates

For each polysemous word w_i in an instance, we try to find a support term $w_j | i \neq j$ which suggests a different sense of w_i. We perform Babelfy [4], a word sense disambiguation (WSD) tool, to label the sense of every word in the instance. If w_i is the pun, then s_i, the sense labeled on w_i, is either a common sense or a specific one. These two cases are dealt with separately as follows.

Table 1: Distribution of the Pun Types

Type	Tra.	Pho.	Com.	Sus.	Total
Count	100	45	35	48	228
Ratio	44%	20%	15%	21%	100%

*Tra.=Tractable,Pho.=Phonological, Com.=Compound, Sus.=Suspicious

In the first case, the WSD tool assigns a common sense s_i to w_i based on the context of w_i. If w_i is the pun, there exists another sense s_i', and a support word $w_j | i \neq j$ having a high similarity between s_j and s_i'. We first measure the similarities between each possible sense of w_i and the sense of each word in the context, and then get s_i', the other sense of w_i, which has the maximum similarity to s_j, the sense of another word in the context.

$$s_i' = \underset{s_i^k \in w_i | s_i^k \neq s_i}{\operatorname{argmax}} \max_{j|j \neq i} Sim(s_i^k, s_j)$$

where $Sim(s_i^k, s_j)$ is measured with cosine similarity using SenseEmbed [1]. Finally, we suggest w_i as a pun candidate if it has the highest similarity change defined below, which is also the score of this candidate.

$$SimChange_{w_i} = \frac{exp(Sim_{w_i}^{spec} - Sim_{w_i}^{com})}{exp(Sim_{w_i}^{com})}$$

where $Sim_{w_i}^{spec}$ and $Sim_{w_i}^{com}$ are the scores of the specific and the common meanings of w_i, respectively.

$$Sim_{w_i}^{spec} = \max_{j|j \neq i} Sim(s_i', s_j), \quad Sim_{w_i}^{com} = \max_{j|j \neq i} Sim(s_i, s_j)$$

In the second case, the WSD tool assigns a specific sense s_i to the w_i mainly because the decision of WSD is affected by the support term in the context. Thus, we expect that the decision will be changed if the support term is replaced. For w_i, we look up its support term w_j, which has the highest similarity between s_i and s_j. Then, w_j is replaced with an out-of-vocabulary (OOV) word, and we perform WSD again on the revised instance. If the newly labeled sense s_i' differs from s_i, then w_i will be suggested as a pun candidate with the score $Sim(s_i, s_j)$.

If no pun candidate is found, all polysemous words in the instance will be suggested as the pun candidates with score 1. This is also our baseline method for comparison.

3.2 Selection of the Pun

We propose three strategies to pick up the best one from the pun candidates nominated by the first stage. The first strategy chooses the pun candidate with the highest score. In the tie-condition, one of the top candidates will be randomly chosen. The second strategy randomly selects a candidate which locates in the rear half part of the instance. The third strategy picks up the last candidate in the instance.

4. EXPERIMENTS

For evaluation, a collection of punny jokes are obtained from the Pun of the Day website (www.punoftheday.com). We randomly select 228 instances and annotate the location of the pun for each instance. We also categorize pun instances into four types: Phonological, Compound, Suspicious, and Tractable. The Phonological type indicates the instance has a homophonic pun such as (S4), which is out of the scope of this paper. The instance of Compound may have multiple puns or one pun in multiple-word form. For example, the bolded fragments in "*Wally wanted a career with a big hamburger chain, but he* ***got into a pickle*** *when he couldn't* ***cut the mustard***" are two puns in multiple-word form. Suspicious stands for the cases in which a pun is hardly found.

Table 2: Results of the Pun Candidate Nomination

Model	Precision	Recall	F-Score
Baseline	16.6%	98.0%	28.0%
Our method	21.2%	88.0%	33.9%

Table 3: Overall Accuracy of Pun Location

	Selection		
Nomination	Max-Score	Second-Half	Last
Baseline	18.4%	34.5%	57.0%
Our method	32.0%	41.5%	64.0%

In the example "*Hard water is sometimes used to make soft drinks*", the punster plays on the opposite meanings of the words *hard* and *soft*, but this joke has only one interpretation and hence is not a real pun. This work focuses on the major type Tractable, in which the puns are in single-word form and homographic. The distribution of the four pun types is shown in Table 1. The 100 instances of the Tractable type are our test data.

Table 2 shows the results of pun candidate nomination. The baseline method achieves a very high recall because it keeps nearly all possible terms. Our method achieves a relatively lower recall, but the F-score is higher than the baseline method due to the higher precision. In other words, we surely narrow down the feasible candidates. The accuracies of the combinations of the two nomination methods and the three selection strategies are reported in Table 3. The third selection strategy, which selects the last candidate, performs well with both nomination methods. The best combination achieves an accuracy of 64.0%.

Our method relies on the information provided by Babelfy and SensEmebed, both of which are based on Wikipedia and WordNet. However, a pun may use a slang meaning which is not covered in Babelfy and SensEmbed. The other source of errors is that the support term may be in a multiple-word form. Our word-based similarity measurement performs less accurately in this case.

5. CONCLUSIONS

This paper introduces a novel framework for homographic pun location identification. Based on our observations, we propose various strategies for nomination and selection of the pun candidates. Experimental results not only show the effectiveness of our approach, but also confirm our observations on the support term and the position of puns.

6. ACKNOWLEDGMENTS

This research was supported by Ministry of Science and Technology, Taiwan, under grant 104-2221-E-002-061-MY3.

7. REFERENCES

[1] I. Iacobacci, M. T. Pilehvar, and R. Navigli. Sensembed: Learning sense embeddings for word and relational similarity. In *Proceedings of ACL 2015*, pages 95–105, July 2015.

[2] A. Jaech, R. Koncel-Kedziorski, and M. Ostendorf. Phonological pun-derstanding. In *Proceedings of NAACL HLT 2016*, pages 654–663, June 2016.

[3] T. Miller and I. Gurevych. Automatic disambiguation of english puns. In *Proceedings of ACL 2015*, pages 719–729, July 2015.

[4] A. Moro, A. Raganato, and R. Navigli. Entity Linking meets Word Sense Disambiguation: a Unified Approach. *Transactions of the Association for Computational Linguistics (TACL)*, 2:231–244, 2014.