Joint Detection and Location of English Puns

Yanyan Zou and **Wei Lu**
StatNLP Research Group
Singapore University of Technology and Design
`yanyan_zou@mymail.sutd.edu.sg, luwei@sutd.edu.sg`

Abstract

A pun is a form of wordplay for an intended humorous or rhetorical effect, where a word suggests two or more meanings by exploiting polysemy (*homographic pun*) or phonological similarity to another word (*heterographic pun*). This paper presents an approach that addresses pun detection and pun location jointly from a sequence labeling perspective. We employ a new tagging scheme such that the model is capable of performing such a joint task, where useful structural information can be properly captured. We show that our proposed model is effective in handling both homographic and heterographic puns. Empirical results on the benchmark datasets demonstrate that our approach can achieve new state-of-the-art results.

1 Introduction

There exists a class of language construction known as *pun* in natural language texts and utterances, where a certain word or other lexical items are used to exploit two or more separate meanings. It has been shown that understanding of puns is an important research question with various real-world applications, such as human-computer interaction (Morkes et al., 1999; Hempelmann, 2008) and machine translation (Schröter, 2005). Recently, many researchers show their interests in studying puns, like detecting pun sentences (Vadehra, 2017), locating puns in the text (Cai et al., 2018), interpreting pun sentences (Sevgili et al., 2017) and generating sentences containing puns (Ritchie, 2005; Hong and Ong, 2009; Yu et al., 2018). A pun is a wordplay in which a certain word suggests two or more meanings by exploiting polysemy, homonymy, or phonological similarity to another sign, for an intended humorous or rhetorical effect. Puns can be generally categorized into two groups, namely hetero-graphic puns (where the pun and its latent target are phonologically similar) and homographic puns (where the two meanings of the pun reflect its two distinct senses) (Miller et al., 2017). Consider the following two examples:

(1) When the church bought gas for their annual barbecue, proceeds went from the sacred to the *propane*.

(2) Some diets cause a *gut* reaction.

The first punning joke exploits the sound similarity between the word "*propane*" and the latent target "profane", which can be categorized into the group of heterographic puns. Another categorization of English puns is homographic pun, exemplified by the second instance leveraging distinct senses of the word "*gut*".

Pun detection is the task of detecting whether there is a pun residing in the given text. The goal of pun location is to find the exact word appearing in the text that implies more than one meanings. Most previous work addresses such two tasks separately and develop separate systems (Pramanick and Das, 2017; Sevgili et al., 2017). Typically, a system for pun detection is built to make a binary prediction on whether a sentence contains a pun or not, where all instances (with or without puns) are taken into account during training. For the task of pun location, a separate system is used to make a single prediction as to which word in the given sentence in the text that trigger more than one semantic interpretations of the text, where the training data involves only sentences that contain a pun. Therefore, if one is interested in solving both problems at the same time, a pipeline approach that performs pun detection followed by pun location can be used.

Compared to the pipeline methods, joint learning has been shown effective (Katiyar and Cardie, 2016; Peng et al., 2018) since it is able to re-

duce error propagation and allows information exchange between tasks which is potentially beneficial to all the tasks. In this work, we demonstrate that the detection and location of puns can be jointly addressed by a single model. The pun detection and location tasks can be combined as a sequence labeling problem, which allows us to jointly detect and locate a pun in a sentence by assigning each word a tag. Since each context contains a maximum of one pun (Miller et al., 2017), we design a novel tagging scheme to capture this structural constraint. Statistics on the corpora also show that a pun tends to appear in the second half of a context. To capture such a structural property, we also incorporate word position knowledge into our structured prediction model. Experiments on the benchmark datasets show that detection and location tasks can reinforce each other, leading to new state-of-the-art performance on these two tasks. To the best of our knowledge, this is the first work that performs joint detection and location of English puns by using a sequence labeling approach.[1]

2 Approach

2.1 Problem Definition

We first design a simple tagging scheme consisting of two tags $\{N, P\}$:

- N tag means the current word is <u>n</u>ot a pun.
- P tag means the current word is a <u>p</u>un.

If the tag sequence of a sentence contains a P tag, then the text contains a pun and the word corresponding to P is the pun.

The contexts have the characteristic that each context contains a maximum of one pun (Miller et al., 2017). In other words, there exists only one pun if the given sentence is detected as the one containing a pun. Otherwise, there is no pun residing in the text. To capture this interesting property, we propose a new tagging scheme consisting of three tags, namely $\{B, P, A\}$.

- B tag indicates that the current word appears <u>b</u>efore the pun in the given context.
- P tag highlights the current word is a <u>p</u>un.
- A tag indicates that the current word appears <u>a</u>fter the pun.

We empirically show that the BPA scheme can guarantee the context property that there exists a maximum of one pun residing in the text.
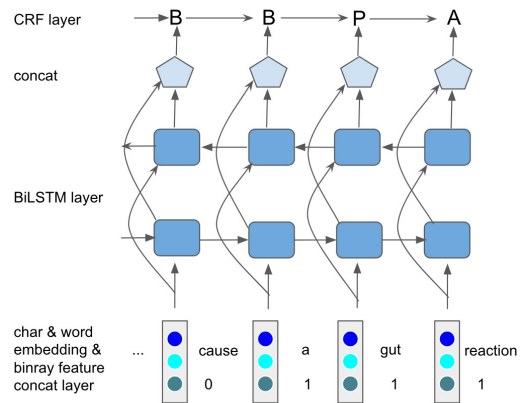

Figure 1: Model architecture

Given a context from the training set, we will be able to generate its corresponding gold tag sequence using a deterministic procedure. Under the two schemes, if a sentence does not contain any puns, all words will be tagged with N or B, respectively. Exemplified by the second sentence "*Some diets cause a gut reaction*," the pun is given as "*gut*." Thus, under the BPA scheme, it should be tagged with P, while the words before it are assigned with the tag B and words after it are with A, as illustrated in Figure 1. Likewise, the NP scheme tags the word "*gut*" with P, while other words are tagged with N. Therefore, we can combine the pun detection and location tasks into one problem which can be solved by the sequence labeling approach.

2.2 Model

Neural models have shown their effectiveness on sequence labeling tasks (Chiu and Nichols, 2016; Ma and Hovy, 2016; Liu et al., 2018). In this work, we adopt the bidirectional Long Short Term Memory (BiLSTM) (Graves and Schmidhuber, 2005) networks on top of the Conditional Random Fields (Lafferty et al., 2001) (CRF) architecture to make labeling decisions, which is one of the classical models for sequence labeling. Our model architecture is illustrated in Figure 1 with a running example. Given a context/sentence $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ where n is the length of the context, we generate the corresponding tag sequence $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ based on our designed tagging schemes and the original annotations for pun detection and location provided by the corpora. Our model is then trained on pairs of $(\mathbf{x}, \mathbf{y})$.

Input. The contexts in the pun corpus hold the property that each pun contains exactly one content word, which can be either a noun, a verb, an

adjective, or an adverb. To capture this characteristic, we consider lexical features at the character level. Similar to the work of (Liu et al., 2018), the character embeddings are trained by the character-level LSTM networks on the unannotated input sequences. Nonlinear transformations are then applied to the character embeddings by highway networks (Srivastava et al., 2015), which map the character-level features into different semantic spaces.

We also observe that a pun tends to appear at the end of a sentence. Specifically, based on the statistics, we found that sentences with a pun that locate at the second half of the text account for around 88% and 92% in homographic and heterographic datasets, respectively. We thus introduce a binary feature that indicates if a word is located at the first or the second half of an input sentence to capture such positional information. A binary indicator can be mapped to a vector representation using a randomly initialized embedding table (He et al., 2017; Wang and Lu, 2018). In this work, we directly adopt the value of the binary indicator as part of the input.

The concatenation of the transformed character embeddings, the pre-trained word embeddings (Pennington et al., 2014), and the position indicators are taken as input of our model[2].

Tagging. The input is then fed into a BiLSTM network, which will be able to capture contextual information. For a training instance $(\mathbf{x}, \mathbf{y})$, we suppose the output by the word-level BiLSTM is $\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n)$. The CRF layer is adopted to capture label dependencies and make final tagging decisions at each position, which has been included in many state-of-the-art sequence labeling models (Ma and Hovy, 2016; Liu et al., 2018). The conditional probability is defined as:

$$P(\mathbf{y}|\mathbf{x}) = \frac{\prod_{i=1}^{n} \exp\left(W_{y_{i-1}, y_i} \mathbf{z}_i + b_{y_{i-1}, y_i}\right)}{\sum_{\mathbf{y}' \in \mathbf{Y}} \prod_{i=1}^{n} \exp\left(W_{y'_{i-1}, y'_i} \mathbf{z}_i + b_{y'_{i-1}, y'_i}\right)}$$

where $\mathbf{Y}$ is a set of all possible label sequences consisting of tags from $\{N, P\}$ (or $\{B, P, A\}$), W_{y_{i-1}, y_i} and b_{y_{i-1}, y_i} are weight and bias parameters corresponding to the label pair (y_{i-1}, y_i). During training, we minimize the negative log-likelihood summed over all training instances:

$$\mathcal{L} = -\sum_i \log P(\mathbf{y}_i | \mathbf{x}_i)$$

where $(\mathbf{x}_i, \mathbf{y}_i)$ refers to the i-th instance in the training set. During testing, we aim to find the optimal label sequence for a new input $\mathbf{x}$:

$$\mathbf{y}^* = \arg\max_{\mathbf{y} \in \mathbf{Y}} P(\mathbf{y}|\mathbf{x})$$

This search process can be done efficiently using the Viterbi algorithm.

3 Experiments

3.1 Datasets and Settings

We evaluate our model on two benchmark datasets (Miller et al., 2017). The homographic dataset contains 2,250 contexts, 1,607 of which contain a pun. The heterographic dataset consists of 1,780 contexts with 1,271 containing a pun. We notice there is no standard splitting information provided for both datasets. Thus we apply 10-fold cross validation. To make direct comparisons with prior studies, following (Cai et al., 2018), we accumulated the predictions for all ten folds and calculate the scores in the end.

For each fold, we randomly select 10% of the instances from the training set for development. Word embeddings are initialized with the 100-dimensional Glove (Pennington et al., 2014). The dimension of character embeddings is 30 and they are randomly initialized, which can be fine tuned during training. The pre-trained word embeddings are not updated during training. The dimensions of hidden vectors for both char-level and word-level LSTM units are set to 300. We adopt stochastic gradient descent (SGD) (Bottou, 1991) with a learning rate of 0.015.

For the pun detection task, if the predicted tag sequence contains at least one P tag, we regard the output (i.e., the prediction of our pun detection model) for this task as true, otherwise false. For the pun location task, a predicted pun is regarded as correct if and only if it is labeled as the gold pun in the dataset. As to pun location, to make fair comparisons with prior studies, we only consider the instances that are labeled as the ones containing a pun. We report precision, recall and F_1 score in Table 1. A list of prior works that did not employ joint learning are also shown in the first block of Table 1.

3.2 Results

We also implemented a baseline model based on conditional random fields (CRF), where features like POS tags produced by the Stanford POS tagger (Toutanova et al., 2003), n-grams, label tran-

---

[2]The word sense has also been shown helpful for the location of a homographic pun (Cai et al., 2018). However, such information may not always be helpful for the location of heterographic puns. We thus exclude such knowledge.

System	Homographic						Heterographic					
	Detection			Location			Detection			Location		
	P.	*R.*	F_1	*P.*	*R.*	F_1	*P.*	*R.*	F_1	*P.*	*R.*	F_1
Pedersen (2017)	78.32	87.24	82.54	44.00	44.00	44.00	73.99	86.62	68.71	-	-	-
Pramanick and Das (2017)	72.51	90.79	68.84	33.48	33.48	33.48	73.67	94.02	71.74	37.92	37.92	37.92
Mikhalkova and Karyakin (2017)	79.93	73.37	67.82	32.79	32.79	32.79	75.80	59.40	57.47	35.01	35.01	35.01
Vadehra (2017)	68.38	47.23	46.71	34.10	34.10	34.10	65.23	41.78	42.53	42.80	42.80	42.80
Indurthi and Oota (2017)	90.24	89.70	85.33	52.15	52.15	52.15	-	-	-	-	-	-
Vechtomova (2017)	-	-	-	65.26	65.21	65.23	-	-	-	79.73	**79.54**	**79.64**
Cai et al. (2018)	-	-	-	81.50	74.70	78.00	-	-	-	-	-	-
CRF	87.21	64.09	73.89	**86.31**	55.32	67.43	89.56	70.94	79.17	**88.46**	62.76	73.42
Ours – NP	89.19	86.25	87.69	82.11	70.82	76.04	85.33	90.64	87.91	79.17	71.76	75.28
Ours – BPA	89.24	92.28	91.04	83.55	**77.10**	**80.19**	84.62	**95.20**	89.60	81.41	77.50	79.40
Ours – BPA-p	**91.25**	**93.28**	**92.19**	82.06	76.54	79.20	86.67	93.08	**89.76**	80.81	75.22	77.91
Pipeline	-	-	-	67.70	67.70	67.70	-	-	-	68.84	68.84	68.84

Table 1: Comparison results on two benchmark datasets. (*P.*: Precision, *R.*: Recall, F_1: F_1 score.)

sitions, word suffixes and relative position to the end of the text are considered. We can see that our model with the BPA tagging scheme yields new state-of-the-art F_1 scores on pun detection and competitive results on pun location, compared to baselines that do not adopt joint learning in the first block. For location on heterographic puns, our model's performance is slightly lower than the system of (Vechtomova, 2017), which is a rule-based locator. Compared to CRF, we can see that our model, either with the NP or the BPA scheme, yields significantly higher recall on both detection and location tasks, while the precisions are relatively close. This demonstrates the effectiveness of BiLSTM, which learns the contextual features of given texts – such information appears to be helpful in recalling more puns.

Compared to the NP scheme, the BPA tagging scheme is able to yield better performance on these two tasks. After studying outputs from these two approaches, we found that one leading source of error for the NP approach is that there exist more than one words in a single instance that are assigned with the P tag. However, according to the description of pun in (Miller et al., 2017), each context contains a maximum of one pun. Thus, such a useful structural constraint is not well captured by the simple approach based on the NP tagging scheme. On the other hand, by applying the BPA tagging scheme, such a constraint is properly captured in the model. As a result, the results for such a approach are significantly better than the approach based on the NP tagging scheme, as we can observe from the table. Under the same experimental setup, we also attempted to exclude word position features. Results are given by BPA-p. It is expected that the performance of pun location drops, since such position features are able to capture the interesting property that a pun tends

to appear in the second half of a sentence. While such knowledge is helpful for the location task, interestingly, a model without position knowledge yields improved performance on the pun detection task. One possible reason is that detecting whether a sentence contains a pun is not concerned with such word position information.

Additionally, we conduct experiments over sentences containing a pun only, namely 1,607 and 1,271 instances from homographic and heterographic pun corpora separately. It can be regarded as a "pipeline" method where the classifier for pun detection is regarded as perfect.[3] Following the prior work of (Cai et al., 2018), we apply 10-fold cross validation. Since we are given that all input sentences contain a pun, we only report accumulated results on pun location, denoted as Pipeline in Table 1. Compared with our approaches, the performance of such an approach drops significantly. On the other hand, such a fact demonstrates that the two task, detection and location of puns, can reinforce each other. These figures demonstrate the effectiveness of our sequence labeling method to detect and locate English puns in a joint manner.

3.3 Error Analysis

We studied the outputs from our system and make some error analysis. We found the errors can be broadly categorized into several types, and we elaborate them here. 1) Low word coverage: since the corpora are relatively small, there exist many unseen words in the test set. Learning the representations of such unseen words is challeng-

---

[3]Under a pipeline setting, the first step is to detect if a sentence contains a pun. Then another algorithm is called to locate the exact pun word residing in the sentence if such a sentence is detected as the one containing a pun. In our setting, we assume the detection phase is perfect. In other words, all sentences containing a pun are exactly retrieved.

ing, which affects the model's performance. Such errors contribute around 40% of the total errors made by our system. 2) Detection errors: we found many errors are due to the model's inability to make correct pun detection. Such inability harms both pun detection and pun location. Although our approach based on the BPA tagging scheme yields relatively higher scores on the detection task, we still found that 40% of the incorrectly predicted instances fall into this group. 3) Short sentences: we found it was challenging for our model to make correct predictions when the given text is short. Consider the example "*Superglue! Tom rejoined,*" here the word *rejoined* is the corresponding pun. However, it would be challenging to figure out the pun with such limited contextual information.

4 Related Work

Most existing systems address pun detection and location separately. Pedersen (2017) applied word sense knowledge to conduct pun detection. Indurthi and Oota (2017) trained a bidirectional RNN classifier for detecting homographic puns. Next, a knowledge-based approach is adopted to find the exact pun. Such a system is not applicable to heterographic puns. Doogan et al. (2017) applied Google n-gram and word2vec to make decisions. The phonetic distance via the CMU Pronouncing Dictionary is computed to detect heterographic puns. Pramanick and Das (2017) used the hidden Markov model and a cyclic dependency network with rich features to detect and locate puns. Mikhalkova and Karyakin (2017) used a supervised approach to pun detection and a weakly supervised approach to pun location based on the position within the context and part of speech features. Vechtomova (2017) proposed a rule-based system for pun location that scores candidate words according to eleven simple heuristics. Two systems are developed to conduct detection and location separately in the system known as UWAV (Vadehra, 2017). The pun detector combines predictions from three classifiers. The pun locator considers word2vec similarity between every pair of words in the context and position to pinpoint the pun. The state-of-the-art system for homographic pun location is a neural method (Cai et al., 2018), where the word senses are incorporated into a bidirectional LSTM model. This method only supports the pun location task on ho-

mographic puns. Another line of research efforts related to this work is sequence labeling, such as POS tagging, chunking, word segmentation and NER. The neural methods have shown their effectiveness in this task, such as BiLSTM-CNN (Chiu and Nichols, 2016), GRNN (Xu and Sun, 2016), LSTM-CRF (Lample et al., 2016), LSTM-CNN-CRF (Ma and Hovy, 2016), LM-LSTM-CRF (Liu et al., 2018).

In this work, we combine pun detection and location tasks as a single sequence labeling problem. Inspired by the work of (Liu et al., 2018), we also adopt a LSTM-CRF with character embeddings to make labeling decisions.

5 Conclusion

In this paper, we propose to perform pun detection and location tasks in a joint manner from a sequence labeling perspective. We observe that each text in our corpora contains a maximum of one pun. Hence, we design a novel tagging scheme to incorporate such a constraint. Such a scheme guarantees that there is a maximum of one word that will be tagged as a pun during the testing phase. We also found the interesting structural property such as the fact that most puns tend to appear at the second half of the sentences can be helpful for such a task, but was not explored in previous works. Furthermore, unlike many previous approaches, our approach, though simple, is generally applicable to both heterographic and homographic puns. Empirical results on the benchmark datasets prove the effectiveness of the proposed approach that the two tasks of pun detection and location can be addressed by a single model from a sequence labeling perspective.

Future research includes the investigations on how to make use of richer semantic and linguistic information for detection and location of puns. Research on puns for other languages such as Chinese is still under-explored, which could also be an interesting direction for our future studies.

References

Léon Bottou. 1991. Stochastic gradient learning in neural networks. In *Proc. of Neuro-Nimes*.

Yitao Cai, Yin Li, and Xiaojun Wan. 2018. Sense-aware neural models for pun location in texts. In *Proceedings of ACL*.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Proceedings of TACL*.

Samuel Doogan, Aniruddha Ghosh, Hanyang Chen, and Tony Veale. 2017. Idiom savant at semeval-2017 task 7: Detection and interpretation of english puns. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Alex Graves and Jürgen Schmidhuber. 2005. Frame-wise phoneme classification with bidirectional lstm networks. In *Proceedings of 2005 IEEE International Joint Conference on Neural Networks*, volume 4, pages 2047–2052.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of ACL*.

Christian F Hempelmann. 2008. Computational humor: Beyond the pun? *The Primer of Humor Research. Humor Research*, 8:333–360.

Bryan Anthony Hong and Ethel Ong. 2009. Automatically extracting word relationships as templates for pun generation. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*.

Vijayasaradhi Indurthi and Subba Reddy Oota. 2017. Fermi at semeval-2017 task 7: Detection and interpretation of homographic puns in english language. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Arzoo Katiyar and Claire Cardie. 2016. Investigating lstms for joint extraction of opinion entities and relations. In *Proceedings of ACL*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*.

Liyuan Liu, Jingbo Shang, Xiang Ren, Frank F Xu, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower sequence labeling with task-aware neural language model. In *Proceedings of AAAI*.

Fuli Luo, Tianyu Liu, Zexue He, Qiaolin Xia, Zhifang Sui, and Baobao Chang. 2018. Leveraging gloss knowledge in neural word sense disambiguation by hierarchical co-attention. In *Proceedings of EMNLP*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of ACL*.

Elena Mikhalkova and Yuri Karyakin. 2017. Pun-fields at semeval-2017 task 7: Employing roget's thesaurus in automatic pun recognition and interpretation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Tristan Miller, Christian Hempelmann, and Iryna Gurevych. 2017. Semeval-2017 task 7: Detection and interpretation of english puns. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

John Morkes, Hadyn K Kernal, and Clifford Nass. 1999. Effects of humor in task-oriented human-computer interaction and computer-mediated communication: A direct test of srct theory. *Human-Computer Interaction*, 14(4):395–435.

Ted Pedersen. 2017. Duluth at semeval-2017 task 7: Puns upon a midnight dreary, lexical semantics for the weak and weary. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Hao Peng, Sam Thomson, Swabha Swayamdipta, and Noah A. Smith. 2018. Learning joint semantic parsers from disjoint data. In *Proceedings of NAACL*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*.

Aniket Pramanick and Dipankar Das. 2017. Ju cse nlp @ semeval 2017 task 7: Employing rules to detect and interpret english puns. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Graeme Ritchie. 2005. Computational mechanisms for pun generation. In *Proceedings of the Tenth European Workshop on Natural Language Generation (ENLG-05)*.

Thorsten Schröter. 2005. *Shun the Pun, Rescue the Rhyme?: The Dubbing and Subtitling of Language Play in Film*. Ph.D. thesis, Estetisk-filosofiska fakulteten.

Özge Sevgili, Nima Ghotbi, and Selma Tekir. 2017. N-hance at semeval-2017 task 7: A computational approach using word association for puns. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387*.

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of NAACL-HLT*.

Ankit Vadehra. 2017. Uwav at semeval-2017 task 7: Automated feature-based system for locating puns. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Olga Vechtomova. 2017. Uwaterloo at semeval-2017 task 7: Locating the pun using syntactic characteristics and corpus-based metrics. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Bailin Wang and Wei Lu. 2018. Learning latent opinions for aspect-level sentiment classification. In *Proceedings of AAAI*.

Jingjing Xu and Xu Sun. 2016. Dependency-based gated recursive neural network for chinese word segmentation. In *Proceedings of ACL*.

Zhiwei Yu, Jiwei Tan, and Xiaojun Wan. 2018. A neural approach to pun generation. In *Proceedings of ACL*.