# WECA: A WordNet-Encoded Collocation-Attention Network for Homographic Pun Recognition

**Yufeng Diao[1,2], Hongfei Lin[1]*, Di Wu[1], Liang Yang[1], Kan Xu[1],**
Zhihao Yang[1], Jian Wang[1], Shaowu Zhang[1], Bo Xu[1], DongyuZhang[1]
[1]DaLian University of Technology, Da Lian, China
[2]Inner Mongolia University for Nationalities, Tong Liao, China
diaoyufeng@mail.dlut.edu.cn, hflin@dlut.edu.cn
wudi@dlut.edu.cn, liang@dlut.edu.cn
xukan@dlut.edu.cn, yangzh@dlut.edu.cn
wangjian@dlut.edu.cn, zhangsw@dlut.edu.cn
xubo2011@mail.dlut.edu.cn, zhangdongyu@dlut.edu.cn

## Abstract

Homographic puns have a long history in human writing, widely used in written and spoken literature, which usually occur in a certain syntactic or stylistic structure. How to recognize homographic puns is an important research. However, homographic pun recognition does not solve very well in existing work. In this work, we first use WordNet to understand and expand word embedding for settling the polysemy of homographic puns, and then propose a WordNet-Encoded Collocation-Attention network model (WECA) which combined with the context weights for recognizing the puns. Our experiments on the SemEval2017 Task7 and Pun of the Day demonstrate that the proposed model is able to distinguish between homographic pun and non-homographic pun texts. We show the effectiveness of the model to present the capability of choosing qualitatively informative words. The results show that our model achieves the state-of-the-art performance on homographic puns recognition.

## 1 Introduction

A pun is a writers use of a word in an ambiguous and inconsistent way in language, often to play on the different meanings of the word or utilize similarly pronounced sounds for a common humorous effect. Puns are widely used in written and spoken literature, which intended as jokes. For example, Tom Swifty by(Lippman and Dunn, 2000), in which puns usually occur in a certain syntactic or stylistic structure. From literature, speeches and oral storytelling, puns are also a standard rhetorical device, which also can be applied non-humorously. For instance, Shakespeare is well known for his puns, which continually appeared in his non-comedic works by (Tanaka, 1992). Both humorous and non-humorous puns have been the theme of extensive and attractive works that has led to discernment for the nature of puns with double meaning.

There are many relevant studies on pun recognition in natural language processing. Many scholars attempted to classify puns according to the similar relationship between the pronunciations and double meanings of the words. For example, (Pafford, 1987) categorizes pun into homophonic puns and homographic puns, which used homonyms and polysemy of words respectively. The research on pun recognition has carried out according to this classification system of Redfern. Our work also considers that puns consist of homophonic and homographic puns.

| Type of Pun | Example | Pun Word |
|---|---|---|
| Homographic | I used to be a banker but I lost interest. | interest |
| Homophonic | When the church bought gas for their annual barbecue, proceeds went from sacred to the propane. | propane |

Table 1: Pun Examples

Both homographic puns and homophonic puns have double meanings to increase deep impression in a certain environment. However, two types of puns have their own features, respectively. Homographic puns, as an important class of puns, which the words for two senses of puns share the same orthographic form. While homophonic puns have the similarity in pronunciations with double senses that distinguished from homographic puns. The former one mainly settles synonyms, while the latter one solves homonyms. Because of the difference, we can not use the unified model to distinguish. Table 1 illustrates the examples of homographic pun and homophonic pun.

In this study, we mainly focus on homographic

*Corresponding author



*Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2507–2516
Brussels, Belgium, October 31 - November 4, 2018. 

puns since they widely used everywhere (Miller, Tristan and Turković, Mladen, 2016) and easily obtain in existing corpus. However, homographic puns recognition in the current works does not solve very well because of their confused double meanings.

To solve the mentioned problem, we propose a computational WordNet-Encoded Collocation-Attention network model (WECA) to recognize homographic puns. Our model takes semantic word embedding and collocation into account for homographic puns recognition. Based on the experiments, the results show that our work will improve the performance of homographic puns recognition. This work is the first to recognize homographic puns with improved word representation and attention mechanism to the best of knowledge. Here, our contributions are as follows.

- The paper applies the lexical ontology WordNet to understand and extend the word embedding for solving the polysemy of homographic puns.
- The paper proposes a neural attention mechanism to extract the collocation for homographic puns classification, which combined with Bi-LSTM to obtain the context weights.
- Experimental results on the datasets of Semeval2017 Task7 and Pun of the Day demonstrate our method outperforms several baselines for recognition homographic puns. Furthermore, visualization of selected examples show the reasons that this model works well.

The rest of this paper is structured in the following. Section 2 mainly reviews the related work on word representation and puns classification. Section 3 presents our proposed word embedding and collocation attention-based network model. Section 4 shows our experiments and discusses evaluation results. Finally, Section 5 concludes our research contributions and offers the future work.

## 2 Related Work

In this section, we will review related works on word representation and homographic pun recognition for homographic puns classification briefly.

### 2.1 Word Representation

In recent years, word representation has the great improvement because it solves data sparsity problem and obtain more semantic relations between words compared with one-hot representation.

(Rumelhart et al., 1986) proposed the idea of word distributed representation, which converts all the words into a low-dimensional continuous semantic space. This space took each word as a vector. These distributed low-dimensional word representation have been widely applied in many NLP tasks, including machine translation(Sutskever et al., 2014; Bahdanau et al., 2014), text classification (Niu et al., 2017; Du et al., 2017), neural language models (Mikolov et al., 2010, 2013) and parsing (Chen and Manning, 2014; Chen et al., 2015). Word embedding is taken as the essential and available inputs for NLP tasks, which enables encoding semantic representation in meaningful vector space.

The studies show that word representations are useful to achieve a good balance between effectiveness and efficiency, such as Word2Vec (Mikolov et al., 2013) and GloVe(Pennington et al., 2014). Therefore, the semantic meanings of words can reflect in the contexts according to these distributed representation models.

However, homographic puns always have multiple meanings. The word representation, considering as only one vector for each word, which puzzled by the understanding for polysemy of puns. This paper combines the representations of lemmas, synsets and words from WordNet[1](Miller, 2002) to understand multiple meanings of homographic puns. The lemma and synset annotation in WordNet provide helpful semantic information for detecting homographic puns.

### 2.2 Homographic Pun Recognition

In recent years, homographic puns have increasingly become a respectable research topic, which widely appears in rhetoric and literary criticism. However, there were little related works in the fields of computational linguistics and natural language processing by (Miller, Tristan and Turković, Mladen, 2016). In this subsection, we mainly introduce some puns detecting methods.

There are many useful methods to classify the puns in NLP. For example, (Kao et al., 2016; Huang et al., 2017) used a probability statistical model to capture the latent semantic information between words for detecting homographic puns. (Jaech et al., 2016) proposed a new prob-

[1]WordNet: http://wordnet.princeton.edu/

ability model to learn phoneme edit probabilities for classifying the homophonic puns. The system ECNU(Xiu et al., 2017) applied a supervised training classifier, which helpful features derived from WordNet and Word2Vec embeddings to distinguish between homographic puns. The system Fermi (Indurthi and Oota, 2017) employed a supervised approach for the detection of homographic puns. It used a bi-directional RNN for a classification model and adopted the distributed semantic word embeddings as input features. These methods do not consider the collocation between words in homographic puns.

The attention mechanism proposed by (Bahdanau et al., 2014) to settle machine translation problem, which was used to select the reference words for words before translation. (Xu et al., 2015) used attention model for image generation to select the similar image regions. For text classification, (Yang et al., 2016) applied attention mechanism into solving document-level classification. Many other tasks in NLP used this mechanism, including natural language question answering (Kumar et al., 2015), parsing (Vinyals et al., 2014), image question answering(Yang et al., 2015), and classification(Shen et al., 2018; Tan et al., 2018). Therefore, this model is capable of discovering the important and semantic information. Meanwhile, attention mechanism can also improve the performance of classification tasks.

Hence, we explore an attention mechanism for collocation to mine the latent semantic information between the part of speech words to achieve good result for homographic puns recognition.

## 3 Methods

Homographic puns recognition could influence by considering both the semantic word embedding and collocation with the context weights for homographic puns. In this section, we propose our model as WordNet-Encoded Collocation-Attention network (WECA). Figure 1 demonstrates the overall structure of our model.

It consists of three main components: an improved word embedding with WordNet-Encoded as inputs, a Bidirectional Long Short-Term Memory (Bi-LSTM) as context weights in a sentence for homographic puns and a fully-connected network as the collocation-attention mechanism. The attention networks combined by a concatenate operation to discover the collocation. Then the context weights and attention networks combined by an element-wise multiplication operation in the classification layer. We describe the details of three components as follows.

### 3.1 WordNet-Encoded Word Embedding

The homographic pun is a clever trick to let one word relate to two aspects or multiple meanings. For example, "Before he sold Christmas trees, he got himself spruced up" The pun word spruced has two meaning: one meaning is spruce tree, while the other is making oneself or something look neater and tidier. We find this word spruced means the last meaning in this situation. Therefore, the polysemy of the ambiguity from homographic puns need additional large lexical ontology. Thus, we apply WordNet for computational linguistics and natural language processing.

Polysemy is critical factor for recognizing homographic puns. To combine the information of multiple meanings, we propose giving a WordNet-Encoded model (WE) to obtain the word embedding for each word. WordNet is a lexical ontology of words. Each word has multiple semantics corresponding with respect to different senses and each sense corresponds to multiple words.

We introduce lemmas, synsets (senses) and words in WordNet. For example, the word is "interest". The word "interest" has three main synsets: sake (a reason for wanting something done), pastime (a diversion that occupies one's time and thoughts) and interest (a sense of concern with and curiosity about someone or something). The lemmas eliminate the ambiguity of each sense. For instance, the synset pastime represents a diversion that occupies one's time and thoughts, which contains lemmas pastime, interest and pursuit. Then we propose two strategies to generate the Word-Net-Encoded embedding based on the information of lemmas and synsets information, Average Lemma Aggregation Model (ALA) and Weighted Lemma Aggregation Model (WLA).

**Average Lemma Aggregation Model** (ALA) adopts a strategy of equal weight according to meanings of homographic puns. ALA model mixes all the lemmas of all the senses of a word together for each word. Hence, it represents the target word by using the average of its whole lemma embedding and puts this together on the original

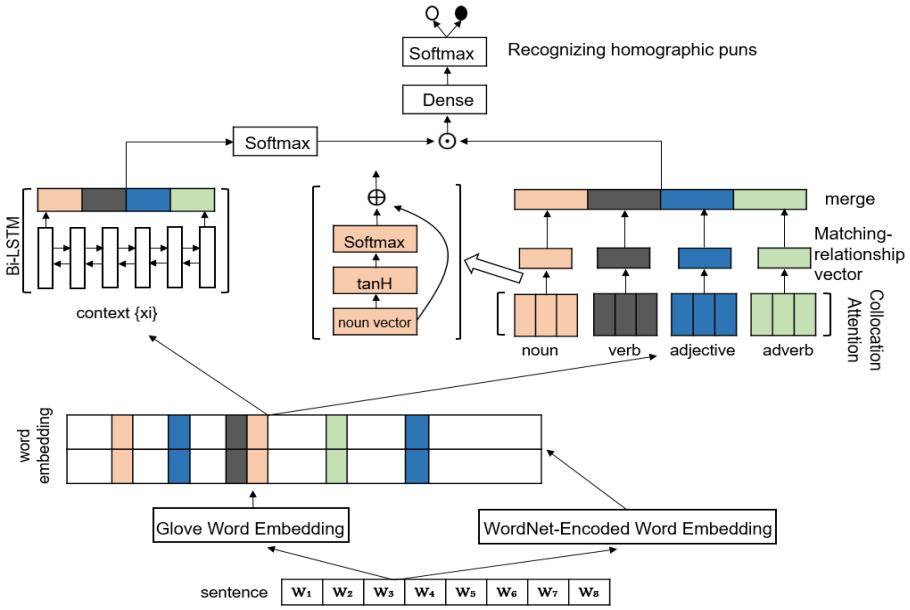


Figure 1: WordNet-Encoded Collocation-Attention network model(WECA)

vector of target word. The formula is as follows:

$$w = \frac{1}{m} \sum_{s_i{}^{(w)} \in S^{(w)}} \sum_{l_i{}^{(s_j)} \in L_i{}^{(s_j)}} w^{l_j s_j} \quad (1)$$

which means the new embedding vector of $w$ is determined by the average of all its lemma embedding. Here, $m$ represents the number of lemmas with overlapping senses with respect to the word $w$, $s_i$ is the sense $i$, $l_j$ is the lemma $j$. Finally, word embedding of $w$ is the concatenation of the original vector and above new vector.

ALA model can apply lemmas to encoding latent semantic relationship because lemmas share the information by multiple words and senses. Therefore, words sharing the same lemmas are likely to obtain the similar representations.

**Weighted Lemma Aggregation Model** (WLA) The ALA Model takes the lemma embedding to encode lemma information for word representation. Although ALA model represents the average of all the lemma, which does not consider the importance of certain lemmas. Hence, we construct embedding for a target word with the help of word senses and lemmas in WordNet that we called WLA model. The formula is as follows:

$$w = \sum_{s_i{}^{(w)} \in S^{(w)}} \frac{|L_i{}^{(s_i)}|}{m} \sum_{l_i{}^{(s_j)} \in L_i{}^{(s_j)}} w^{l_j s_j} \quad (2)$$

where $m$ represents for all the number of lemmas with overlapping senses with respect to the word $w$, $s_i$ is the sense $i$, $l_j$ is the lemma $j$, $l_i{}^{(s_j)}$ is the number of lemmas in each sense with target word, $w$ is the new embedding considering the weighted lemma information. Then, the target word embedding concatenates new vector to original vector.

The weighted lemma strategy assumes one sense of word obtains more attention if this sense of word has more lemmas. We can show each word as a special distribution on the sense. From the results, WLA model is the best representation.

### 3.2 Bidirectional Long Short Term Memory(Bi-LSTM) for Recognizing Homographic Puns

Long Short Term Memory (LSTM) was proposed by Hochreiter and Schmiduber (1997), which has been widely adopted for text processing. There are three gates and one cell in LSTM: an input gate $i_t$, a forget gate $f_t$, an output gate $o_t$ and a memory cell $c_t$. They are all vector in $R^d$. The equations of transition are:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1}) \quad (3)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + V_f c_{t-1}) \quad (4)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + V_o c_{t-1}) \quad (5)$$

$$\tilde{c_t} = tanh(W_c x_t + U_c h_{t-1}) \quad (6)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{7}$$

$$h_t = o_t \odot tanh(c_t) \tag{8}$$

where $x_t$ is an input vector at the current time step, $\sigma$ is the sigmoid function and $\odot$ is the element-wise multiplication operation, $W_{\{i,f,o,c\}}$ ,$U_{\{i,f,o,c\}}$ ,$V_{\{i,f,o,c\}}$ are learned weight parameters, $h_t$ is the hidden state vector. In LSTM, the hidden state $h_t$ only encodes the front context in a forward direction but not consider the backward context.

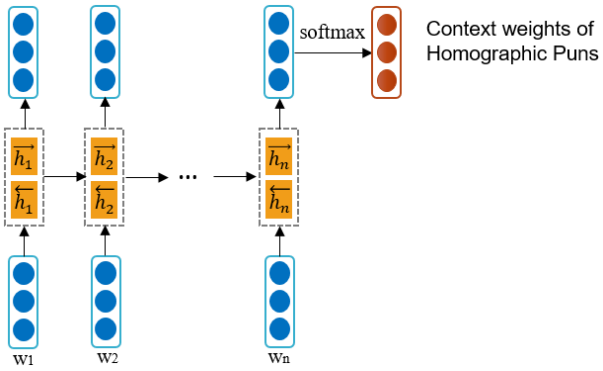


Figure 2: The Structure of Bi-LSTM

In this study, we apply Bi-LSTM model(Graves, 2012) to capture the latent semantic information of homographic puns for obtaining the context weights. For each sentence, it has a forward LSTM $\overrightarrow{h}$ and a backward LSTM $\overleftarrow{h}$ to concatenate the hidden states of two LSTMs as the representation of corresponding word. Figure 2 illustrates the architecture of Bi-LSTM model. $\{w_1, w_2, \cdots, w_n\}$ represent the word vector in a sentence whose length is $N$. Then, the forward and backward contexts can take into account simultaneously. The equations of transition are:

$$\overrightarrow{h}_t = H(W_{x\overrightarrow{h}} x_t + W_{\overrightarrow{h}\overrightarrow{h}} \overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}) \tag{9}$$

$$\overleftarrow{h}_t = H(W_{x\overleftarrow{h}} x_t + W_{\overleftarrow{h}\overleftarrow{h}} \overleftarrow{h}_{t-1} + b_{\overleftarrow{h}}) \tag{10}$$

$$h_{out} = W_{\overrightarrow{h}y} \overrightarrow{h}_t + W_{\overleftarrow{h}y} \overleftarrow{h}_t + b_y \tag{11}$$

where $\overrightarrow{h}_t$ is a forward LSTM, $\overleftarrow{h}_t$ is a backward LSTM, $h_{out}$ is the output of Bi-LSTM.

### 3.3 Collocation-Attention Mechansim

It proposes that not all the words provide the same contribution of word representation for the sentence. Especially for homographic puns recognition, the collocation between candidate pun words in a sentence offers more clues for getting the collocational word weights. Miller points out that the candidate pun words mainly consist of nouns, verbs, adjectives and adverbs in each pun. For example, "The money doesn't grow on the tree, but it can grow on the branch." The word "branch" is the pun word. From this example, we know that the collocation of candidate pun words {money, grow, tree, branch}, which should be more important for recognizing homographic puns.

Therefore, it is necessary to learn about the latent relationship in collocation of words. We design an attention mechanism to obtain the collocational weights by extracting such words of collocation from nouns, verbs, adjectives and adverbs, respectively. Then we concentrate on the four parts in sentences with pun to aggregate the informative words for classifying the homographic puns. This model uses an attention network taking word embedding with WordNet-Encoded as input then to extract polysemy attention signal, which made use of polysemy to understand ambiguity of homographic puns. The formula is as follows:

$$u_{ijt} = V \cdot tanh(W_u h_{ijt} + b_w) \tag{12}$$

$$\alpha_{ijt} = \frac{exp(u_{ijt})}{\sum_{t=1}^{T_x} exp(u_{ijt})} \tag{13}$$

$$c_{ij} = \sum_{t=1}^{T_x} \alpha_{ijt} h_{ijt} \tag{14}$$

where $h_{ijt}$ is a hidden state at each time step for each part of speech, $j \in \{nouns, verbs, adjectives, adverbs\}$, $u_{ijt}$ is a hidden representation of $h_{ijt}$ through a one-layer MLP, $\alpha_{ijt}$ is a normalized importance weight through a softmax function with each part of speech, $c_{ij}$ is a context vector as a high level representation over the words from attention-based model by the weighted mean of the hidden state sequence $h_{ijt}$ for each part of speech.

After combining the attention networks with the context weights in a sentence, we merge all the $c_{ij}$ vectors from collocation attention model and take the uniformed context weights in a sentence.

Then we mix the two parts results with element wise multiplication operation to recognize the homographic puns. The formula is as follows:

$$c_i = [c_{inouns}; c_{iverbs}; c_{iadjectives}; c_{iadverbs}] \quad (15)$$

$$l_{out} = Softmax(h_{out}) \quad (16)$$

$$s_i = c_i \cdot l_{out} \quad (17)$$

where $c_i$ is merged by $c_{ij}$, $j \in \{nouns, verbs, adjectives, adverbs\}$, $l_{out}$ is the softmax function of $h_{out}$, $s_i$ is the result with the multiplication operation of $c_i$ and $l_{out}$.

The model can be trained in an end-to-end way by backpropagation, where objective function is the cross-entropy loss. Let $y$ be the target distribution and $\hat{y}$ be the predicted distribution. The goal of training is to minimize the cross-entropy error between $y$ and $\hat{y}$ for all sentences.

$$loss = -\sum_i \sum_j y_i^j log\hat{y}_i^j + \lambda\|\theta\|^2 \quad (18)$$

where $i$ is the index of sentence, $j$ is the index of class. Our classification is two way. $\lambda$ is the L2 regularization term. $\theta$ is the parameter set.

## 4 Experiments and Evaluation

In this section, we first evaluate the effectiveness of our WordNet-Encode model (WE) on two tasks to detect the polysemy of homographic puns. Then, we examine the performance of our WECA model compared with existing methods. Finally, we show the effectiveness of our model.

### 4.1 Experimental Setting

In this section, we introduce datasets, evaluation metrics, baseline methods, and present the details of the training process of our model.

**Datasets** To verify the effectiveness of our proposed model, we use two datasets: SemEval2017 Task7[2] and Pun of the Day[3] .

**SemEval2017 Task7.** This dataset is composed of homographic and heterographic puns for recognizing and interpreting puns. We focus on homographic puns detection in semantic rather than phonology. The homographic pun word will have at least two words sense in the WordNet(Miller and Gurevych, 2015). Table 1 shows a detailed statistical distribution of our datasets.

**Pun of the Day.** This dataset only includes pun content in the beginning. Then it collects the negative samples from Yahoo! Answer[4] , AP News[5] , Proverb, and New York Times in order to balance the distribution of positive and negative examples, which adapt to decrease the domain discrepancy. Table 2 provides a complete statistical description of our dataset.

| Dataset | Positive | Negative | Average Length |
|---|---|---|---|
| Task7 | 1607 | 643 | 13.1 |
| Pun of the Day | 2423 | 2403 | 13.5 |

Table 2: Statistics of Datasets

**Metrics** We apply the standard measures precision, recall, accuracy and F1-score to evaluate the effectiveness for homographic puns recognition, which also adopted as metrics in SemEval 2017 Task7 evaluation.

**Baselines** We compare several strong baselines as follows.

**LSTM:** LSTM without WordNet-Encoded embedding and Collocation-Attention mechanism.

**Bi-LSTM:** Bi-LSTM without WordNet-Encoded embedding and Collocation-Attention.

**Bi-LSTM$_E$:** Bi-LSTM with WordNet-Encoded embedding used the WLA model.

**Bi-LSTM-Attention:** Bi-LSTM with single attention mechanism.

Fermi and N-Hance are the good performing model in the SemEval2017 task7.

**Top1 Fermi:** Fermi took a supervised approach for homographic puns detection. It did not construct own train data set, but rather split the shared task data set into train sets and test sets(Miller, Tristan and Hempelmann, Christian and Gurevych, Iryna, 2017). It used a Bi-RNN to learn a classification model and treat the word embedding as the input features.

**Top2 N-Hance:** It assumed every pun had a particularly strong association with exactly one other word in context(Miller, Tristan and Hempelmann, Christian and Gurevych, Iryna, 2017). Then it calculated PMI between words in context to detect and locate puns. If the score exceeded

[2]SemEval2017 Task7: htto://alt.qcri.org/semeval2017/task7/
[3]Pun of the Day: htto://www.punoftheday.com/
[4]htto://answers.yahoo.com/
[5]htto://hosted.ap.org/dynamic/fronts/HOME?SITE=AP

a certain threshold, the text assumed to contain a pun. Otherwise, the text assumed to have no pun.

**WECA:** Here, we use Bi-LSTM with WordNet-Encoded embedding with WLA model and Collocation-Attention mechanism.

**Training Details** In experiments, our model is tuned with 5-fold cross validation. All word vectors are initialized by GloVe. We use 50, 100, 200 and 300 dimension to verify the performance, respectively. Here, 200 dimension is the best performance. Therefore, we set the dimensions of word, synset and lemma embedding to be 200. The size of units in LSTM is 800. RMSprop is used for our optimization method. We use learning rate decay and early stop in the training process. All models are trained by mini-batch of 64 instances.

### 4.2 The Effectiveness of WordNet-Encoded Word Embedding

Comparing the GloVe model with our ALA model and WLA model, we evaluate the quality of our improved word representations to detect the homographic puns. In this experiment, we use the same classifier Bi-LSTM and parameters to verify the effectiveness of our word embedding.

Figure 3 and Figure 4 show the results of different word embedding for detecting homographic puns. From the results we can observe that:

(1) Our models ALA and WLA, which outperform the original vector GloVe on both two datasets. It indicates that our model can better capture the semantic relations of words by utilizing lemma annotation properly based on the WordNet.

(2) The ALA model represents each word with the average of its lemma embedding. In general, the ALA model performs better than GloVe, showed that lemma and synset of WordNet is very useful. The reason is the words sharing mutual lemma representation are helpful with each other.

(3) The WLA model mostly performs better than GloVe and ALA model. This model can obtain a weighted distribution according to the senses and lemmas. The results show that their different senses are commonly different from others, but share certain representation.

### 4.3 Pun Recognition with WordNet-Encoded Collocation-Attention Network (WECA)

Our model WECA, combination of WordNet-Encoded word embedding and Collocation-Attention network with context weight, which performs compared with the suggested baselines.

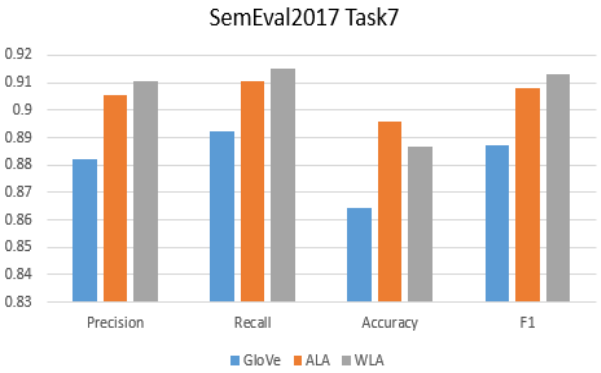


Figure 3: Comparison of Different Word Embedding on SemEval2017 Task7

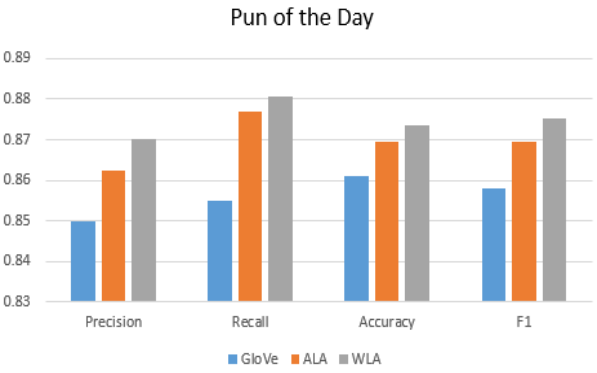


Figure 4: Comparison of Different Word Embedding on Pun of the Day

Here, we use Pun of the Day as the training set to obtain all the parameters, and test the results of homographic pun recognition in SemEval2017 task7. The results are shown in Table 3.

| | Precision | Recall | F1 |
|---|---|---|---|
| LSTM | 81.80 | 83.7 | 82.43 |
| Bi-LSTM | 85.40 | 83.64 | 84.51 |
| Bi-LSTM$_E$ | 85.87 | 85.07 | 85.46 |
| Bi-LSTM-Attention | 84.92 | 85.62 | 85.26 |
| N-Hance | 75.53 | **93.34** | 83.50 |
| WECA | **89.19** | 90.64 | **89.21** |

Table 3: Comparison of Different Models of Homographic Puns Recognition

(1) Bi-LSTM has the better performance for homographic puns detection compared with LSTM (84.51% vs.82.43%). It shows that Bi-LSTM exploits two parallels to discover more context information. At the same time, Bi-LSTM$_E$ outperforms Bi-LSTM (85.46% vs.84.51%), which demonstrating the effectiveness of the WordNet-Encoded word embedding.

(2) Bi-LSTM-Attention performs slightly better

| | Precision | Recall | F1 |
|---|---|---|---|
| Fermi | 90.24 | 89.70 | 89.97 |
| WECA | **91.43** | **90.53** | **90.98** |

Table 4: Comparison of WECA and Fermi of Homographic Puns Recognition

than Bi-LSTM and LSTM (85.26% vs. 84.51%, 82.43%). The reason is that the attention mechanism can assign the weight to the whole words according to the context information.

(3) Our model WECA has a better performance compared with Bi-LSTM$_E$, Bi-LSTM-Attention and N-Hance (87.45% vs. 85.46%, 85.26%, 83.50%). N-Hance is the second place in SemEval2017 task7. It shows the WordNet-Encoded word embedding can capture more semantic information between words with the help of lemma and synsets in WordNet. Meanwhile, it presents that the attention network mechanism combined with collocation of the specific part of speech of puns, which capture the characteristic information to recognize the homographic puns.

The best perform of SemEval2017 task7 is Fermi. However, Fermi only evaluates on 675 of 2250 homographic contexts(Miller, Tristan and Hempelmann, Christian and Gurevych, Iryna, 2017) in SemEval2017 task7. Thus, our model uses 675 as a test set and rest of data as a training set. The results are shown in Table 4. Experiment results present our model outperforms Fermi under the same data distribution. It shows the effectiveness of our model again.

### 4.4 Visualization of Model

In order to verify our model is enable to select the valuable information of words that reflected the collocation, we visualize the attention layers for several sentences in Pun of the Day and SemEval2017 Task7 data sets whose labels are correctly predicted by our model in Figure 5. We choose two examples. One presents collocation between nouns, the other presents collocation between verbs.

Each line is a sentence. Blue denotes word weight. If color of word is darker, the word is more important. Figure 5 shows our WECA model can select words carrying ambiguous meanings from the collocation of homographic puns. For example, in the first sentence, it highlights "interest", which is worthy attracting more attention because of multiple meanings for the pun word. In the second sentence, "sleep" is selected word by our attention model as related to homographic puns. Therefore, attention networks of collocation is effective for recognizing homographic puns.

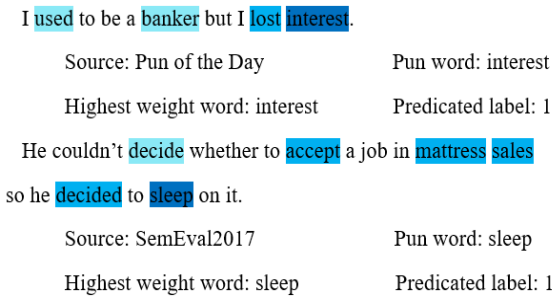


Figure 5: Visualization of attention layers

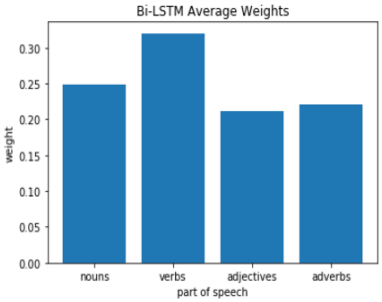


Figure 6: Average context weight of all sentences by Bi-LSTM

We apply the Bi-LSTM to capture latent semantic context for weighting part of speech which included nouns, verbs, adjectives and adverbs from forward and backward direction. Figure 6 shows Bi-LSTM model distributes weights to the four part of speeches. The weights of verbs occupy the first place, then second one is nouns, adjectives and adverbs are lower. Meanwhile, it demonstrates the importance of part of speech.

Thus, we choose two examples to illustrate weights with the four part of speeches according to context information by Bi-LSTM in Figure7. For the first example, the word "cured", as a verb, which is a pun word. It shows weights of verbs are highest allocation by Bi-LSTM. For the second example, the word "cinch", as a noun, which is a pun word. It illustrates that Bi-LSTM distributes the higher weights to nouns, which presents the importance of nouns from the context semantic information. Hence, context weights providing by Bi-LSTM are helpful of the collocation for recognizing the homographic puns.

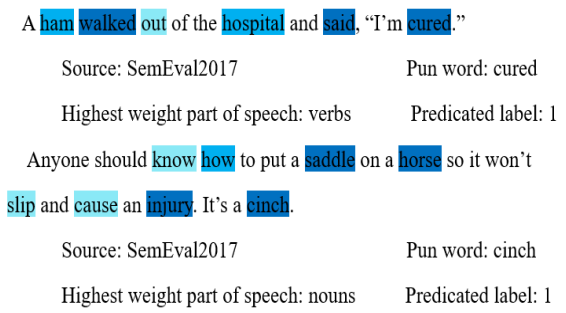


Figure 7: Visualization of Bi-LSTM

## 5 Conclusion and Future Work

In this study, we propose a computational model WECA combined with WordNet-Encoded word embedding and Collocation-Attention network. We extend the semantic information of word embedding by lemma and synset according to WordNet. We also apply a neural attention network, combined with Bi-LSTM, which captures the collocation of homographic puns. Experimental results show our model achieves the best performance and outperforms several baselines.

In future work, we would like to find an appropriate way in incorporating the external linguistic knowledge to improve the performance of homographic puns recognition. We also focus on automatically generating homographic puns. Those are all promising jobs we can pursue in the future.

## Acknowledgements

This work is partially supported by grant from the Natural Science Foundation of China (No.61632011, 61702080, 61572102, 61602079, 61602078), the Fundamental Research Funds for the Central Universities (No.DUT18ZD102, DUT17RC(3)016).

## References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *Computer Science*.

D. Chen and C. D. Manning. 2014. A fast and accurate dependency parser using neural networks. In *Conference on Empirical Methods in Natural Language Processing*, pages 740–750.

Wenliang Chen, Min Zhang, and Yue Zhang. 2015. *Distributed feature representations for dependency parsing*. IEEE Press.

Jiachen Du, Ruifeng Xu, Yulan He, and Lin Gui. 2017. Stance classification with target-specific neural attention. In *Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 3988–3994.

Alex Graves. 2012. *Long Short-Term Memory*. Springer Berlin Heidelberg.

Yu Hsiang Huang, Hen Hsen Huang, and Hsin Hsi Chen. 2017. Identification of homographic pun location for pun understanding. In *International Conference on World Wide Web Companion*, pages 797–798.

Vijayasaradhi Indurthi and Subba Reddy Oota. 2017. Fermi at semeval-2017 task 7: Detection and interpretation of homographic puns in english language. In *International Workshop on Semantic Evaluation*, pages 457–460.

Aaron Jaech, Rik Koncel-Kedziorski, and Mari Ostendorf. 2016. Phonological pun-derstanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 654–663.

Justine T. Kao, Roger Levy, and Noah D. Goodman. 2016. A computational model of linguistic humor in puns. *Cognitive Science*, 40(5):1270–1285.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2015. Ask me anything: dynamic memory networks for natural language processing. pages 1378–1387.

L. G. Lippman and M. L. Dunn. 2000. Contextual connections within puns: effects on perceived humor and memory. *Journal of General Psychology*, 127(2):185–197.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Computer Science*.

Tomas Mikolov, Martin Karafit, Lukas Burget, Jan Cernock, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010, Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September*, pages 1045–1048.

G. A. Miller. 2002. Wordnet: A lexical database for the english language. *Contemporary Review*, 241(1):206–208.

Tristan Miller and Iryna Gurevych. 2015. Automatic disambiguation of english puns. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 719–729.

Miller, Tristan and Hempelmann, Christian and Gurevych, Iryna. 2017. Semeval-2017 task 7: Detection and interpretation of english puns. In *International Workshop on Semantic Evaluation*, pages 58–68.

Miller, Tristan and Turković, Mladen. 2016. Towards the automatic detection and identification of english puns. *The European Journal of Humour Research*, 4(1):59–75.

Yilin Niu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Improved word representation learning with sememes. In *Meeting of the Association for Computational Linguistics*, pages 2049–2058.

J. H. P. Pafford. 1987. Redfern, w., puns. *Notes Queries*, (3).

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *Parallel Distributed Processing Explorations in the Microstructure of Cognition*, 323(6088):399–421.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. Disan: Directional self-attention network for rnn/cnn-free language understanding. In *AAAI*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. 4:3104–3112.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep semantic role labeling with self-attention. In *AAAI*.

Keiko Tanaka. 1992. The pun in advertising: A pragmatic approach. *Lingua*, 87(1-2):91–102.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2014. Grammar as a foreign language. *Eprint Arxiv*, pages 2773–2781.

Yuhuan Xiu, Man Lan, and Yuanbin Wu. 2017. Ecnu at semeval-2017 task 7: Using supervised and unsupervised methods to detect and locate english puns. In *International Workshop on Semantic Evaluation*, pages 453–456.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. *Computer Science*, pages 2048–2057.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. 2015. Stacked attention networks for image question answering. pages 21–29.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.