# BiLSTM-Based Models for Metaphor Detection

Shichao Sun and Zhipeng Xie

Shanghai Key Laboratory of Data Science
School of Computer Science, Fudan University, China
xiezp@fudan.edu.cn

**Abstract.** Metaphor is a pervasive phenomenon in our daily use of natural language. Metaphor detection has been playing an important role in a variety of NLP tasks. Most existing approaches to this task rely heavily on the use of human-crafted features built from linguistic knowledge resource, which greatly limits their applicability. This paper presents four BiLSTM-based models for metaphor detection. The first three models use a sub-sequence as the input to BiLSTM network, each with a special kind of sub-sequence extracted from the input sentence. The last model is an ensemble model which aggregate the outputs from the first three models to get the final output. Experimental results have shown the effectiveness of our models.

**Keywords:** Metaphor detection, BiLSTM, sub-sequence

## 1 Introduction

Metaphor is a form of figurative language, where sentences consist of words or phrases deviating from their proper definitions in ways that do not permit a literal interpretation. From a cognitive point of view, metaphor can be thought of as a type of conceptual mappings[10] where abstract concepts are mapped to concrete concepts in order to make the readers understand some characteristics of abstract concepts more easily. For example, the utterance "*Time is money*"[10] emphasizes that the abstract "*time*" is valuable just like the concrete "*money*". Metaphors are pervasive in natural language. Corpus studies reveal that metaphors appear averagely in every third sentence of general-domain text. In order to interpret the meanings of sentences containing metaphorical expressions, it is necessary to discriminate the metaphorical and the literal use of languages. Due to the prevalence and importance of metaphorical language, effective detection of metaphors is of great value in a variety of practical NLP applications, such as machine translation, information retrieval, opinion mining and so on.

Previous approaches to metaphor detection can be broadly classified into two categories. The one is to detect metaphor in a single sentence that contains a metaphor[16, 17], and the other is in the discourse[12, 9, 7]. Our approach belongs to the first category. Particularly, given an input sentence, the task is to judge whether a target verb in the sentence is in its metaphorical or literal use.

Such kind of metaphors accounts for a substantial proportion of all metaphorical expressions, approximately 60%[13]. As for this task, most of the previous approaches have used a variety of semantic and syntactic features (such as abstractness, imageability, supersenses and so on), which are normally extracted from some external linguistic knowledge resources. We shall introduce some representative approaches in the Section 2.

Although these existing approaches have been proved to be able to achieve good performance, their reliance on external linguistic resources has greatly limited their applicability, especially to those languages that are lack of linguistic resources.

To mitigate this shortcoming, we propose to use deep neural networks to train an end-to-end model for metaphor detection. In our approach, BiLSTM network is used to build up the feature vector representations of sentences automatically, without any use of linguistic knowledge resources, except an unsupervised text corpus that are commonly available for most languages. The BiLSTM network is chosen because it is a kind of recurrent neural network with the ability to deal with variable length sequence data, and it has been proved to be effective in a lot of NLP tasks [2, 3].

For a given input sentence with a target verb, there are multiple ways to extract a sub-sequence from the sentence, with respect to the target verb. Using different sub-sequences of the sentence as input may affect the performance of BiLSTM model. In other words, different sub-sequences could lead to different feature representation of the sentence via the BiLSTM model. Therefore, we present to extract three kinds of sub-sequences for a given sentence with a target verb. Each kind of sub-sequences corresponds to a single sub-sequence model. In addition, we also propose an ensemble model in order to integrate all these three kinds of sub-sequences together, with the expectation that they can be complementary to each other.

To demonstrate the efficacy of our approaches, we evaluate our model on the metaphor test dataset which was made available by Tsvekov et al.[17]. The dataset is a multilingual test set but we only use the English section. This section contains 111 metaphorical sentences and 111 literal sentences, so it is a balanced dataset.

Our work makes the following contributions: (1) it is the first time that an end-to-end neural model gets proposed to metaphor detection; (2) Our approach is independent of any external linguistic knowledge resources, except pretrained word embeddings on unsupervised text corpus; (3) Three kinds of sub-sequences are presented to be extracted from an input sentence with a target verb, and BiLSTM network is proposed to model these sub-sequences.

## 2 Related Work

Research in automatic metaphor detection has a very long history, ranging from ruled-based methods by using lexical resources to statistical machine learning models. If you want a more thorough review of metaphor processing systems, we

refer the readers to the reviews by Shutova[14]. Here we focus only on the recent approaches using the statistical learning method which often treat metaphor detection as a binary classification problem. For a complete survey, please refer to Shutova[14].

Turney et al. [16] viewed metaphor as a method for transferring knowledge from a familiar, well-understood, or concrete domain to an unfamiliar, less understood, or more abstract domain. They hypothesized that metaphorical word usage is correlated with the degree of abstractness of the word's context. Based on this hypothesis, they used logistic regression algorithm on a feature vector constructed from the abstractness of context words, to classify a word sense in a given context as either literal or metaphorical. In their work, the abstractness ratings for words were calculated automatically by a supervised learning model trained on the MRC database [1].

Heintz et al. [6] applied Latent Dirichlet Allocation (LDA) topic modeling [1] to the problem of automatic extraction of linguistic metaphor and achieved good performance. The hypothesis behind their approach is that a sentence which contains both source and target domain vocabulary could use metaphor, so a sentence which contains different topic may use metaphor. Their result of LDA is 100 topics, where each topic is a probability distribution over the training corpus vocabulary.

Tsvetkov et al. [17] constructed a English metaphor detection system that uses a random forest classifier with conceptual semantic features such as abstractness, imageability, and semantic supersenses. They also supported the hypothesis that metaphors are conceptual, rather than lexical, in nature by showing that their Englished-trained model can detect metaphors in Spanish, Farsi, and Russian.

Klebanov et al. [9] tried to classify each content-word token in a text as a metaphor or non-metaphor. They used various features such as unigrams, part-of-speech, concreteness and topic models to train logistic regression classifier for metaphor detection in running text. The experimental results showed that these features are useful for metaphor detection respectively. It was also shown that the unibram features contribute the most, and the second most effective feature set are the topic models.

Jang et al. [7] hypothesized that topic transition patterns between sentences containing metaphors and their contexts are different from that of literal sentences. They also observed that metaphor is often used to express speaker's emotional experiences and cognitive processes. Therefore, they built up a set of features from the information of sentence-level topic transitions, emotional and cognitive words in metaphorical and literal sentence and their contexts, and used these features to train a support vector machine classifier for metaphor detection.

Most of the existing approaches make use of a variety of features to train classification models for metaphor detection. Most of the features rely on external lexical, syntactic, or semantic linguistic resources that are not always available

[1] http://websites.psychology.uwa.edu.au/school/MRCDatabase/uwa_mrc.htm

(or are expensive to obtain) in all languages, which seriously limits their applicability.

Different from these previous work, this paper presents several metaphor detection models based on BiLSTM neural networks, which are able to build up feature representations of sentences from pretrained word embeddings and do not use any human-crafted features.

## 3 Our Approach

In this paper, we limit our task to determining whether a sentence contains a metaphoric subject-verb-object relation. Different from existing approaches, we propose to use Long-Short Term Memory (LSTM) model that can automatically construct feature representations of sentences for metaphor detection, instead of human-crafted features created from linguistic resources.

Since LSTM is a recurrent neural network model for sequential data, a straightforward solution is to apply LSTM on the original word sequence of the target sentence. Besides this simple solution, we also explore other two possibilities, i.e., applying LSTM on two different sub-sequences extracted from the target sentence (Section 3.1). Correspondingly, three single sub-sequence models based on BiLSTM networks can be constructed, one for each possible kind of sub-sequences, and a fourth model is also proposed as an ensemble model to merge the three single sub-sequence models (Section 3.2). Lastly, we introduce train objective for model(Section 3.3).

### 3.1 Three Kinds of Sub-Sequences

Because LSTM is the recurrent neural network model based on time sequence, different sequences could affect the performance of LSTM model. We want to know how different sub-sequence of the same sentence affect the result of metaphor detection, and find the best sub-sequence to input LSTM model.We propose three sub-sequences of the sentence to input model and then, we will introduce them, respectively.

**The original sequence** is the original form of the sentence in the corpus, and it contains all the information that we need to judge whether the sentence contains the metaphorical use of the target verb or not. This is the longest sub-sequence of the sentence, which will slower the training process of the model. In addition, there may exists some noisy information that possibly affect our judgement.

**The dependency sub-sequence** consists of the target verb, its head word, and all its dependents in the corresponding dependency tree, arranged in their relative positions in the original sentence. In other words, the dependency subsequence contains the words that have direct dependency relationship with the target verb, which is expected to provide most information about whether the

target verb is in metaphorical or literal use. Other syntactic parts, such as the adjective part of the subject or the object, are excluded from the dependency sub-sequence, which are believed to contain little information about our task.

**The SVO sub-sequence** is the sub-sequence consisting of the target verb, its subject, and its object in the sentence, which is a most-common form of syntactic constructions. Compared with the dependency sub-sequence, the SVO sub-sequence is more concise, and excludes adverbial information for metaphor detection. On one hand, shorter SVO sub-sequences can speed up the training process. On the other hand, the adverbial information is discarded, which may be indicative of metaphorical use of the target verb to some degree.

**An illustrative example.** We use a simple example to illustrate the three kinds of sub-sequences. For the original sentence "I have given up smoking for two years.", the dependency tree is shown in**Fig.**1, which is generated automatically by a dependency parser. From the dependency tree, we can easily learn that the dependency sub-sequence is "I have given up smoking for", which consists of all the words that are directly connected to the target verb "given". Then according to the dependency relationship in the dependency tree, we can get the subject "I" and the object "smoking" of the target verb "given", and thus the SVO sub-sequence is "I given smoking". It can be easily seen that all the three kinds of sub-sequences contain the information about the use of the target verb, but with different completeness.

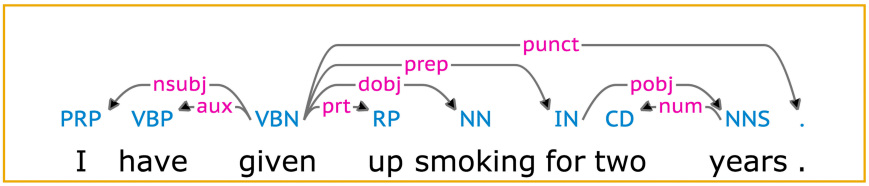


**Fig. 1.** An illustrative dependency tree

In addition, we think three sub-sequences could better adapt to different sentences. As for the original sentence, it may adapt to the sentence that doesnt have any clause, because the sentence is not too long to increase the burden of model training. The dependency sub-sequence may better adapt to the sentence which the predicate is verb phrase, that is the collocation of verb and preposition. Lastly, the SVO could adapt to the sentence which has too many modifiers, because the SVO can use the most most concise sub-sequence to detect metaphor. So we merge three different sequences in order to adapt to all kinds of sentences.

### 3.2 BiLSTM-based Models for Metaphor Detection

Recurrent neural networks (RNNs) are powerful models for variable-length sequence data, and are inherently deep in temporal dimension. Long-short term memory (LSTM) is a popular architecture of RNN, which can mitigate the explosive and vanishing gradient problem.

One shortcoming of LSTM is that it only makes use of its left (or previous) context, and does not utilize its right (or future) context. Bidirectional LSTM (BiLSTM) uses two independent LSTMs to process the sequence on two directions separately, and then concatenate the two final output vectors from both directions.

This section will firstly introduce the basics of LSTM and BiLSTM models, and then propose our BiLSTM-based models for metaphor detection.

**LSTM** is a recurrent neural network with gating mechanism. Here, we adopt the LSTM variant that was introduced by Graves Alex[4]. It comprises four components: an input gate $\mathbf{i}_t$, a forget gate $\mathbf{f}_t$, an output gate $\mathbf{o}_t$, and a memory cell $\mathbf{c}_t$. The formulas for calculating these gate and the memory cell unit are listed as follows:

$$i_t = \sigma_i(x_t W_{xi} + h_{t-1} W_{hi} + w_{ci} \odot c_{t-1} + b_i) \tag{1}$$

$$f_t = \sigma_f(x_t W_{xf} + h_{t-1} W_{hf} + w_{cf} \odot c_{t-1} + b_f) \tag{2}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \sigma_c(x_t W_{xc} + h_{t-1} W_{hc} + b_c) \tag{3}$$

$$o_t = \sigma_o(x_t W_{xo} + h_{t-1} W_{ho} + w_{co} \odot c_t + b_o) \tag{4}$$

The output of LSTM units is the recurrent network's hidden state, which is computed as follows:

$$h_t = o_t \odot \sigma_h(c_t) \tag{5}$$

**BiLSTM** has been proved empirically to be effective in performance improvement over LSTM because it can make use of context in both directions and is thus better in making prediction [2, 3]. **Fig.2** shows the architecture of BiLSTM network.

In our implementation, we process the sequence in two directions, in other words, we process two sequences that the one is the normal sequence, but the other is the reverse sequence. In each direction, all the words $x$ of the sequence will sequentially pass through the LSTM and will get a vector as $h$ which is the feature vector of the sequence which contains the current word and the words before the current word in the sequence, so the last $h$ will become the feature vector of the entire sequence. From two directions, we will get the vector from the normal sequence represented as $\overleftarrow{\mathbf{h}_n}$ and the vector from the reverse sequence represented as $\overrightarrow{\mathbf{h}_n}$. Lastly, we concatenate two vectors of two sequences to form the output of the BiLSTM network represented as $\mathbf{h}$ which could become the feature of the sentence, following the formula:

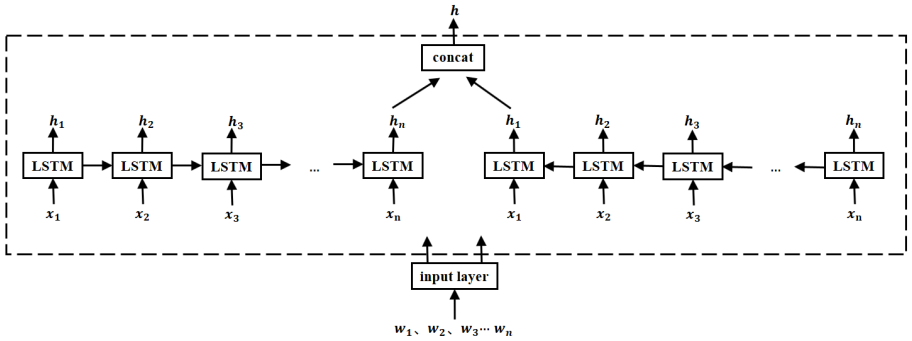


**Fig. 2.** The architecture of BiLSTM

$$\mathbf{X} = [\mathbf{y}] \tag{6}$$

$$\mathbf{h} = [\overrightarrow{\mathbf{h}_n}, \overleftarrow{\mathbf{h}_n}] \tag{7}$$

**The single sub-sequence models** are the models with each kind of sub-sequences as the input to the BiLSTM network. The model consists of three layers : input layer, BiLSTM layer, and softmax layer, as shown in **Fig.3**. In the input layer (also called the embedding layer), each word in the sub-sequence will be converted to an embedding vector by looking up a pretrained word embedding matrix, and then be fed to the BiLSTM layer. After processed by the BiLSTM layer, the information flows to the softmax layer which transforms the output vector from the BiLSTM layer into a probability distribution over the two class labels: *metaphorical* and *literal*.

**The multiple sub-sequences model** can be thought of as an ensemble model, which merges three outputs from the BiLSTM layers in the three single sub-sequence models described above, by using the concatenation layer. where three values of $h$ will be connected to become the feature of the sentence as follows:

$$\mathbf{h} = [\mathbf{h_{sen}}, \mathbf{h_{dep}}, \mathbf{h_{svo}}] \tag{8}$$

then input the result of the concat layer to softmax layer to classify sentence as literal or metaphorical as shown in **Fig.4**. Every value of $h$ is get through the method of the single input model's BiLSTM layer. The value of $h_1$ represents the feature of the original sentence, and the $h_2$ represents the feature of the dependency sub-sequence, lastly, the $h_3$ represents the feature of the SVO sequence.

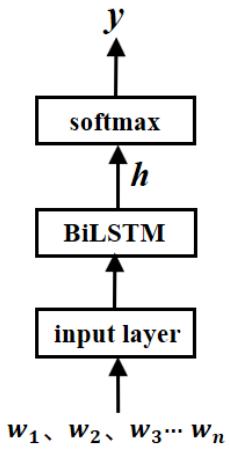


**Fig. 3.** The single sub-sequence model

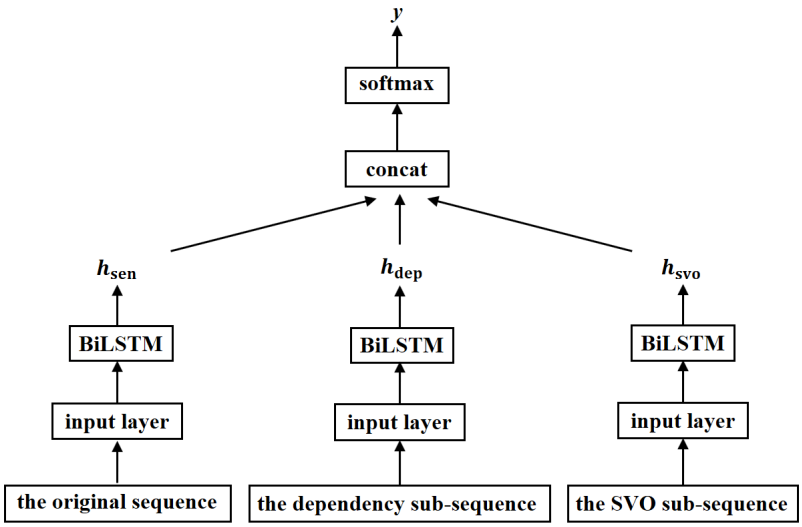


**Fig. 4.** The multiple sub-sequences model

### 3.3 Training Objective

We model the metaphor detection task as a binary classification problem, and use the cross entropy loss function:

$$J = -\frac{1}{N}\sum_{i=1}^{N}(y_i' \log(y_i) + (1 - y_i') \log(1 - y_i)) \tag{9}$$

where $N$ denotes the number of training examples; $y_i'$ is the ground truth label of the $i$-th training example, and $y_i$ is the modelś probability output for the $i$-th training example.

## 4 Experimental Results

In this section, we would like to evaluate the proposed models. firstly, we will introduce the dataset containing the training dataset and test dataset and the data pre-processing(Section 4.1),then we will introduce the experiment particulars containing deep learning tools that we use to implement our model, tricks and the hyper-parameters setting in the model (Section 4.2).Lastly, we will introduce the results in test set (Section 4.3).

### 4.1 Dataset and data pre-processing

**Training set** is from TroFi[2] corpus that consists of 3737 manually annotated English sentences from the Wall Street Journal. Each sentence in TroFi contains either a literal use or a metaphorical use for one of 50 English verbs.

We firstly used Turbo Parser[3] to parse all the sentences into dependency trees, and then extracted the SVO triples and the dependency sub-sequences from the dependency trees according to the dependency relationships. During the parsing process, we filtered the dependency trees and the SVO triples to eliminate parsing-related errors by a blacklist[4] which was provided by Tsvetkov [17] in 2014 and those with verbs which are not in the TroFi verb list.

The final training dataset consists of only the sentences which can be successfully converted into all the three sub-sequence forms without no error. As a result of the pre-processing, there are 1474 metaphorical sentences and 1046 literal sentences left, among which we randomly select 90% as the training set and the remaining 10% as the validation set.

**Test set** is the English section of multilingual test sets[5] which is open by Tsvetkov et al.[17] in 2014. The set contains original sentences forms and their

[2] http://www.cs.sfu.ca/ anoop/students/jbirke/.
[3] http://www.cs.cmu.edu/ ark/TurboParser/.
[4] https://github.com/ytsvetko/metaphor/blob/master/resources/TroFi/.
[5] https://github.com/ytsvetko/metaphor/blob/master/input/.

SVO forms for 111 metaphorical sentences and 111 literal sentences. So we only need to get dependency sub-sequence by the same method which we use to process the training set. The F1-score of the state-of-the-art system on the test set is 0.79.

### 4.2 Model implementation

To implement our model, we use the lasagne[6] which is a lightweight library to build and train neural networks in Theano. It makes common use cases easy, and does not overrate uncommon cases. Besides, it contains the all code implements which we use in our experiment such as LSTM recurrent network, dropout, softmax and so on.

When we train the models, we use the dropout[15] which is a regularization technique for alleviate overfitting in neural networks by preventing complex co-adaptations on training data. We use dropout in input layer and LSTM layer. Besides, we use mini-batch stochastic gradient descent(SGD) with momentum for optimization.

### 4.3 Hyperparameter settings

In our models, the dropout rate is set to 0.6 that is a better value found after repeated attempts. And, the learning rate is set to 0.002, because it can avoid that the optimal solution may be missed in the gradient descent process as a result of big learning rate.

In addition, the number of hidden units in LSTM layer is different in different input sub-sequences, because the different sub-sequence contains different granularity information, more specifically, the feature vector of the sub-sequences is different. The number of hidden units in the original sentence, dependency sub-sequence and SVO is set to 150, 60, 40, respectively when we train the single input model. Correspondingly, the number of hidden units in every LSTM layer of the multiple input model is set to 150, 60, 40.

Lastly but not least, we will introduce how to convert the word in sub-sequence to the word embedding. We do it by using the open pre-trained word embedding library[7] which is used just like a dictionary. The library is made by training the word2vec[8][14] model on Google News corpus (3 billion running words). And the dimensions of the word embedding is 300. Althrough the library contains the most words, very few words cant be found in library. For these words, we will pass them, more unfortunately, if the word is just the verb, we will pass the instance directly.

[6] https://github.com/Lasagne/Lasagne.
[7] https://github.com/mmihaltz/word2vec-GoogleNews-vectors.
[8] https://code.google.com/archive/p/word2vec/.

### 4.4 Results

We train the four proposed models on the training data, respectively. Three single sub-sequence models are for three different kinds of sub-sequences, the other is multiple sub-sequences model. We test the models on the test set and the experiment results are shown in **Table 1**. Tsvetkov et al.[17] which is described in detail in relation to prior work.

**Table 1.** Experimental comparsion of our proposed metaphor detection models.

| **sub-sequence** | **Precision of test set** | $F_1$**-score** |
|---|---|---|
| Original sentence | 75.83% | 0.77 |
| Dependency sub-sequence | 72.71% | 0.76 |
| SVO sub-sequence | 68.33% | 0.74 |
| Multiple sub-sequences model | 76.67% | 0.78 |

It can be observed from **Table 1** that:

- The single sub-sequence model on original sentences has achieved the highest F1-score among all the three single sub-sequence models; while the single sub-sequence model on SVO sub-sequences has got the lowest F1-score. Such observations coincide with our expectation that the adverbial information about the target verb may be helpful to metaphor detection.
- The multiple sub-sequences model outperforms the three single sub-sequence models, which indicates that the three single sub-sequence models are complementary to each other in some degree.
- The multiple sub-sequence model is competitive to the state-of-the-art. However, Tsvetkov et al.[17] achieved the F1-score 0.79 with the help of several conceptual features such as abstractness, imageability, and supersenses, and our appproach only uses word embeddings pretrained on an unsupervised text corpus.

## 5 Conclusions

In this paper, we first propose an end-to-end neural approach for metaphor detection, which does not rely on any external linguistic knowledge resources, except pretrained word embeddings on unsupervised text corpus. We present three kinds of sub-sequences to be extracted from an input sentence with a target verb, and propose to use BiLSTM network to model these sub-sequences. Experimental results have shown the effectiveness of our approach.

**Acknowledgments** This work is partially supported by National High-Tech R&D Program of China (863 Program) (No. 2015AA015404), the 2016 Civil Aviation Safety Capacity Development Funding Project, and the project "Aircraft Operation Resource Data Exchange and Integration". We are grateful to the anonymous reviewers for their valuable comments.

## References


1. Blei David M., Andrew Y. Ng, and Michael I. Jordan. “Latent Dirichlet Allocation”. In Advances in neural information processing systems, 2002, pp. 601–608.
2. Graves Alex, and Jrgen Schmidhuber. “Framewise phoneme classification with bidirectional LSTM and other neural network architectures”. Neural Networks, 2005, 18(5): 602–610.
3. Graves Alex, Abdel-rahman Mohamed, and Geoffrey Hinton. “Speech recognition with deep recurrent neural networks”. The Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 6645–6649.
4. Graves Alex. “Generating sequences with recurrent neural networks”. arXiv preprint arXiv:1308.0850 (2013).
5. Hochreiter Sepp, and Jrgen Schmidhuber. “Long short-term memory”. Neural computation, 1997, 9(8): 1735–1780.
6. Heintz Ilana, Ryan Gabbard, Mahesh Srinivasan, David Barner, Donald S. Black, Marjorie Freedman, and Ralph Weischedel. “Automatic Extraction of Linguistic Metaphor with LDA Topic Modeling”. In Proceedings of the First Workshop on Metaphor in NLP, 2013, pp. 58-66.
7. Jang Hyeju, Yohan Jo, Qinlan Shen, Michael Miller, Seungwhan Moon, and Carolyn Penstein Ros. “Metaphor Detection with Topic Transition, Emotion and Cognition in Context”. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, 2016, pp. 216–225.
8. Klebanov Beata Beigman, Chee Wee Leong, Michael Heilman, and Michael Flor. “Different texts, same metaphor: unigrams and beyond”. In Proceedings of the Second Workshop on Metaphor in NLP, 2014, pp. 11–17.
9. Klebanov Beata Beigman, Chee Wee Leong, and Michael Flor. “Supervised word-level metaphor detection: Experiments with concreteness and reweighting of examples”. In Proceedings of the Third Workshop on Metaphor in NLP, 2015, pp. 11–20.
10. Lakoff, George, and Mark Johnson. “Metaphors we live by”. University of Chicago press, 2008.
11. Mikolov Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. “Distributed representations of words and phrases and their compositionality”. In Advances in neural information processing systems, 2013, pp. 3111–3119.
12. Marc Schulder and Eduard Hovy. “Metaphor detection through term relevance”. In Proceedings of the Second Workshop on Metaphor in NLP,2014, pp. 18–26.
13. Shutova Ekaterina, and Simone Teufel. “Metaphor corpus annotated for source-target domain mappings”. In Proceedings of LREC, 2010, pp. 3255–3261.
14. Shutova Ekaterina. “Design and Evaluation of Metaphor Processing Systems”. Computational Linguistics, 2015, 41(4): 579–623.
15. Srivastava Nitish, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. “Dropout: a simple way to prevent neural networks from overfitting”. Journal of Machine Learning Research, 2014, 15(1): 1929–1958.
16. Turney Peter D., Yair Neuman, Dan Assaf, and Yohai Cohen. “Literal and metaphorical sense identification through concrete and abstract context”. In Proceedings of the 2011 Conference on the Empirical Methods in Natural Language Processing, pp. 680–690.
17. Tsvetkov Yulia, Leonid Boytsov, Anatole Gershman, Eric Nyberg, and Chris Dyer. “Metaphor detection with cross-lingual model transfer”. In Proceedings of the Annual Meeting of the Association for Computational Linguistics. 2014, pp. 248–258.