Neural Metaphor Detecting with CNN-LSTM Model

Chuhan Wu[1], **Fangzhao Wu**[2], **Yubo Chen**[1], **Sixing Wu**[1],
Zhigang Yuan[1] **and Yongfeng Huang**[1]
[1]Tsinghua National Laboratory for Information Science and Technology,
Department of Electronic Engineering, Tsinghua University, Beijing 100084
[2]Microsoft Research Asia
{wuch15,ybch14,wu-sx15,yuanzg14,yfhuang}@mails.tsinghua.edu.cn
wufangzhao@gmail.com

Abstract

Metaphors are figurative languages widely used in daily life and literatures. It's an important task to detect the metaphors evoked by texts. Thus, the metaphor shared task is aimed to extract metaphors from plain texts at word level. We propose to use a CNN-LSTM model for this task. Our model combines CNN and LSTM layers to utilize both local and long-range contextual information for identifying metaphorical information. In addition, we compare the performance of the softmax classifier and conditional random field (CRF) for sequential labeling in this task. We also incorporated some additional features such as part of speech (POS) tags and word cluster to improve the performance of model. Our best model achieved 65.06% F-score in the *all POS testing* subtask and 67.15% in the *verbs testing* subtask.

1 Introduction

A metaphor is a type of conceptual mapping to represent one thing as another (Lakofi and Johnson, 1980). They are widely used in verbal and written languages to convey rich linguistic and sentiment information (Steen et al., 2010). Detecting the metaphors in texts are important to mine the semantic and sentiment information better, which is beneficial to many applications such as machine translation, dialog systems and sentiment analysis (Tsvetkov et al., 2014).

However, detecting metaphors is a challenging task. The semantic differences between metaphorical and non-metaphorical texts are often subtle. For example, the sentence *Her hair is a white snowflake* is metaphorical, while the sentence *Her hair is white* doesn't contain metaphors. In addition, detecting metaphors can be influenced by subjective factors, and may need specific domain knowledge (Tsvetkov et al., 2014).

Existing computational approaches to detect metaphors are mainly based on lexicons (Mohler et al., 2013; Dodge et al., 2015) and supervised methods (Turney et al., 2011; Heintz et al., 2013; Klebanov et al., 2014, 2015, 2016). Lexicon-based methods are free from data annotation, but they are unable to detect novel metaphorical usages and capture the contextual information. Supervised methods such as logistic regression classifier (Klebanov et al., 2014) can capture richer metaphor information. However, they need sophisticated hand-crafted features.

To improve the collective techniques on detecting metaphors, the metaphor shared task[1] aims to detect both metaphorical verbs and metaphors with other POS. Given a sentence and their words with specific POS tags, systems are required to determine whether each word is a metaphor. We propose a CNN-LSTM model with CRF or weighted softmax classifier to address this task. Our model can take advantage of both long-range and local information by utilizing both LSTM and CNN layers. We propose to use a weighted softmax classifier to predict the label sequence of sentence, which outperforms the CRF method. We apply a model ensemble strategy to help our model predict more accurately. In addition, we incorporated additional features such as POS tags and word cluster features to further improve our model. Our best model achieved 65.06% F-score on the test data in the *all POS testing* subtask, and 67.15% in the *verbs testing* subtask.

2 CNN-LSTM Model with CRF or Softmax Inference

We model this task as a sequential labeling task and the input is a sentence with a sequence of words. The framework of our CNN-LSTM model

[1]https://competitions.codalab.org/competitions/17805

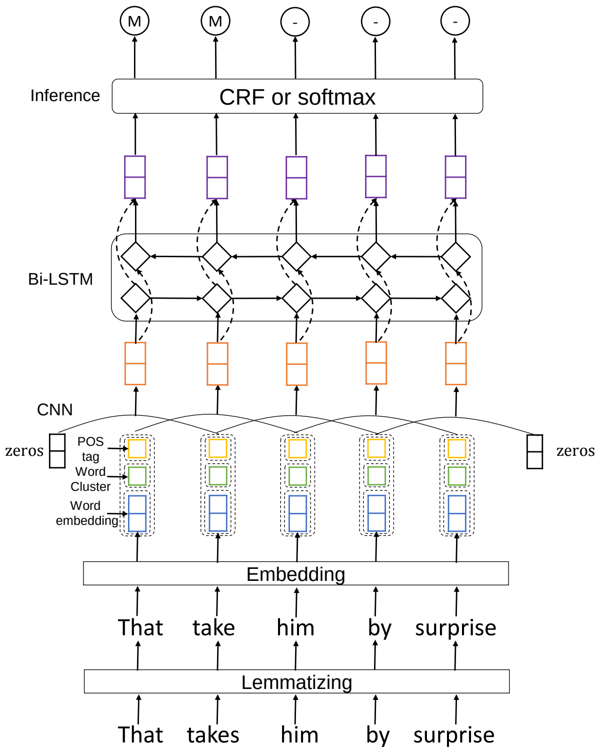

Figure 1: The architecture of our method. The final metaphor labels will be predicted by a CRF or softmax inference layer.

is presented in Figure 1. We will introduce the details of modules in our model from bottom to top.

We follow the approach proposed by Klebanov et al. (2016) to use the lemmatizing strategy. The first module in our model is a lemmatizer. This module is used to lemmatize the verbs in texts via a dictionary. The input is a text with a sequence of word, and output is the text with lemmatized words. Since verbs with different forms can share the same lemmas, using the lemmatized verbs in texts can simplify the semantic information and reduce the number of out-of-vocabulary words. We use the NLTK package (Bird et al., 2009) to transform the verbs into their lemmas.

The second module is an embedding layer. It will convert sequences of words in sentences into sequences of low-dimension dense vectors via a lookup table. The embedding weights of words are obtained by the pre-trained word2vec model and they will be fine-tuned during model training. POS tags are useful in metaphor detecting task (Klebanov et al., 2014). Therefore, we also incorporate the one-hot encoded POS tags as additional features into our neural model, and concatenated them with the word embeddings. We use the Stanford parser[2] tool to obtain the POS tag of each word in texts. Since similar words may have similar metaphor information, we also incorporate the word cluster features. They are obtained by clustering the word embedding vectors via k-means method. They are also one-hot encoded and combined with the word embeddings as the final word representations to input the neural network.

The third module in our model is a convolutional neural networks (CNN) to extract local contextual information. Motivated by the multiple kernels CNN used for sequential labeling (Chen et al., 2016), we also apply such CNN with different window sizes to this task.

The fourth module in our model is a bidirectional long short-term memory (Bi-LSTM) layer. This layer is used to extract the long-range information from the CNN feature maps. It will combine the previous and future context information to output the hidden state $\mathbf{h}_i$ at time step i.

The last module is an inference layer. We implement it with two alternatives and compare their performance via experiments.

CRF: We use CRF to predict the metaphor labels of each words. Given the matrix of hidden representations $\mathbf{h} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_N]$, the conditional probability of the output sequence of label $\mathbf{y}$ is formulated as follows:

$$p(\mathbf{y}|\mathbf{h};\theta) = \frac{\prod_{i=1}^{N} \psi(\mathbf{h}_i, y_i, y_{i-1})}{\sum_{\mathbf{y}' \in \mathcal{Y}(s)} \prod_{i=1}^{N} \psi(\mathbf{h}_i, y'_i, y'_{i-1})}, \quad (1)$$

where $\mathcal{Y}(s)$ is the set of all possible label sequences, θ is the parameters, and $\psi(\mathbf{h}_i, y_i, y_{i-1})$ is the potential function. In our model, we use a simple potential function which is formulated as:

$$\psi(\mathbf{h}_i, y_i, y_{i-1}) = \exp(y_i^T \mathbf{W}^T \mathbf{h}_i + y_{i-1}^T \mathbf{T} y_i), \quad (2)$$

where $\mathbf{W}$ and $\mathbf{T}$ represent the linear transform parameters. The CRF loss function we use is the negative log-likelihood over all training samples, which is formulated as follows:

$$\mathcal{L}_{CRF} = -\sum_{s \in \mathcal{S}} \log(p(\mathbf{y}_s|\mathbf{h}_s;\theta)), \quad (3)$$

where $\mathcal{S}$ is the training set, and $\mathbf{h}_s$ and $\mathbf{y}_s$ are the hidden states and label sequence of sentence s.

[2]https://nlp.stanford.edu/software/lex-parser.shtml

Softmax: We use a dense layer with softmax activation function to predict the metaphor label sequences. Motivated by the cost-sensitive cross-entropy (Santos-Rodríguez et al., 2009; Yang et al., 2014; Muller et al., 2014), the loss function of our model is formulated as follows:

$$\mathcal{L}_{Softmax} = -\sum_{s \in \mathcal{S}} \sum_{i=1}^{N} w_{y_i} y_i \log(\hat{y}_i), \qquad (4)$$

y_i is the metaphor label of i_{th} word, $\hat{y}_i$ is the predicted score, and w_{y_i} is the loss weight of metaphor label y_i. Since there are much more non-metaphorical words than metaphors, we assign larger loss weight to the positive class. Since the prediction is generated from the lemmatized texts, optimizing the loss in Eq. (4) can tune all parameters in the embedding, CNN and LSTM layers.

Ensemble strategy is usually useful to improve the performance of neural network (Wu et al., 2017). We train our model for 20 times on randomly selected 90% training data. For CRF-based model, the prediction of each token will be obtained by voting. For softmax-based model, the output probability is the averaged logits of all model predictions.

3 Experiment

3.1 Dataset and Experimental Settings

The dataset for this task is the VU Amsterdam Metaphor Corpus (VUA)[3]. There are 12,122 sentences for training, and 4,080 sentences for test. We tune the hyper-parameters of our model via cross validation.

The pre-trained word embeddings are the 300-dim Google embedding[4] released by Mikolov et al. (2013). They were trained by the skip-gram model on about 100-billion words on Google News. These word embedding were fine-tuned during model training.

The hyper-parameters in our model were tuned via cross-validation. The dimension of Bi-LSTM hidden states is 200, the window sizes of CNN filters are 3, 5, 7 and 9 respectively. The number of CNN filters is 100. We set the dropout rate to 0.2 for each layer. The loss weights w_p and w_n of metaphors and non-metaphorical words are set to 2.0 and 1.0 respectively. The class number of word cluster is set 50. The batch size is 50, and the max training epoch is set to 15. The optimizer we use is RMSProp in our experiment. The performance of both *all POS testing* and *verbs testing* subtasks is evaluated by precision, recall and F-score as a standard binary classification task.

[3]http://ota.ahds.ac.uk/headers/2541.xml

[4]https://code.google.com/archive/p/word2vec/

3.2 Performance Evaluation

We compare the performance of the variants of our model and several baseline methods. The methods to be compared include: 1) CNN+CRF, using CNN to extract local information and CRF for word-level metaphor detection; 2) *LSTM+CRF*, using Bi-LSTM to obtain the text representation and CRF inference layer; 3) *CNN+LSTM+CRF*, using the combination of LSTM, CNN and CRF inference layer; 4) *CNN+LSTM+CRF+ensemble*, adding ensemble strategy to the CNN+LSTM+CRF model; 5) *CNN+Softmax*, using CNN and weighted softmax classifier for sequential labeling; 6) *LSTM+Softmax*, using Bi-LSTM and softmax inference layer; 7) *CNN+LSTM+Softmax w/o lemma*, using the combination of LSTM, CNN and softmax inference layer, but without the lemmatizing process; 8) *CNN+LSTM+Softmax*, using the combination of LSTM, CNN and softmax inference layer; 9) *CNN+LSTM+Softmax+ensemble*, adding ensemble strategy to the CNN+LSTM+Softmax model. Our official submissions are obtained by model 3), 4), 8), 9) and the different combinations of additional features, which will be discussed in the next subsection.

According to Table 1, we have several observations: (1) The combination of LSTM and CNN outperforms the single CNN and LSTM in both subtasks. It proves that the combination of CNN and LSTM can help to mine both local and long-distance information from texts, which is beneficial for detecting the metaphors in texts. (2) Comparing the modeling using CRF and softmax layer, best precision score can be achieved by using CRF. But the recall and F-score are significantly better when using weighted softmax classifier. This is probably because the numbers of metaphors are usually less than normal non-metaphorical words. The metaphors can be identified better when they are assigned larger loss weights. (3) Improvement can be brought by the lemmatizing process

Model	Verbs Testing			All POS Testing		
	P	R	F	P	R	F
*CNN+CRF**	.628	.611	.619	.605	.589	.597
*LSTM+CRF**	.633	.609	.621	.604	.586	.595
CNN+LSTM+CRF	.644	.615	.629	**.617**	.597	.607
CNN+LSTM+CRF+ensemble	**.664**	.626	.645	.610	.627	.619
*CNN+Softmax**	.575	.716	.638	.585	.644	.613
*LSTM+Softmax**	.588	.710	.643	.591	.659	.623
*CNN+LSTM+Softmax w/o lemma**	.585	.702	.638	.601	.669	.633
CNN+LSTM+Softmax	.593	.734	.656	.611	.677	.643
CNN+LSTM+Softmax+ensemble	.600	**.763**	**.671**	.608	**.700**	**.651**

Table 1: The performance of different methods. *The results of these baseline methods were not submitted due to the limited submission time. We evaluate their performance using the labels of testing data after the competition.

in both tasks. It may be because the lemmatized verbal metaphors are more simple, and there will be fewer out-of-vocabulary words in the embedding look-up table. (4) the ensenmble strategy can also help our model identify metaphors more accurately. It validates that using a series of models to predict can reduce the data noise and improve the generalization ability of our model.

3.3 Influence of Additional Features

Features	Verbs Testing			All POS Testing		
	P	R	F	P	R	F
None	.584	.717	.644	.583	.665	.621
+POS	.588	.729	.651	.606	.662	.633
+cluster	.589	.723	.649	.606	.665	.634
+POS+cluster	**.593**	**.734**	**.656**	**.611**	**.677**	**.643**

Table 2: The influence of additional features on our best-performance model.

The influence of the POS tags and word clusters is shown in Table 2. Here we use the *CNN+LSTM+Softmax* model to investigate the influence of features. The results show that both POS tags and word cluster features can help improve the performance of detecting metaphors. It proves that POS tags contain useful information to identify the metaphors, since metaphors usually have specific POS tags and they can be easier to be identified by incorporating POS information. Thus, combing the POS tag features is beneficial. Incorporating the word cluster features is also useful to improve the performance. It may be because words with similar semantic information have some inherent relatedness and they share similar metaphor information. Our model can identify such information better if word cluster features are incorporated. In addition, it can also enrich the information of out-of-vocabulary words, which can improve the generalization ability of our model. Thus, incorporating the word cluster features is also beneficial to detect metaphors.

3.4 Influence of Loss Weight

Since the metaphors are less frequent than normal words, the selection of loss weight is important. We investigate the influence of the loss weight w_p of positive label on the softmax classifier, which is illustrated in Figure 2. The results indicate that using larger w_p can improve the recall score, but the precision will be lower. It proves that controlling the loss weights can improve the F-score performance in this unbalanced classification task. To achieve a better performance, we choose $w_p = 2$ since the F-score performance is best as shown in this figure.

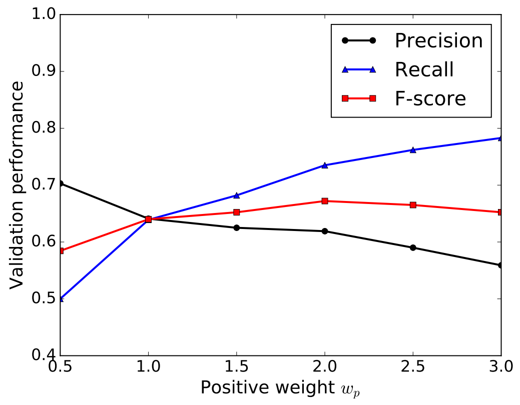

Figure 2: The validation performance of our model using different w_p.

4 Conclusion

In this paper, we introduce our CNN-LSTM model with CRF or softmax layer for the metaphor shared task to detect metaphors in texts. We combine CNN and LSTM to capture both local and long-distance contextual information to represent the input sentences with lemmatizing preprocessing. We compare the performance of using CRF and softmax classifier with weighted loss. In addition, we incorporate additional features including POS tags and word cluster features, and use the ensemble strategy to improve the performance. The experimental results validate the effectiveness of our model on detecting metaphors.

Acknowledgments

The authors thank the reviewers for their insightful comments and constructive suggestions on improving this work. This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB0800402 and in part by the National Natural Science Foundation of China under Grant U1705261, Grant U1536207, Grant U1536201 and U1636113.

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. A feature-enriched neural model for joint chinese word segmentation and part-of-speech tagging. *arXiv preprint arXiv:1611.05384*.

Ellen Dodge, Jisup Hong, and Elise Stickles. 2015. Metanet: Deep semantic automatic metaphor analysis. In *Proceedings of the Third Workshop on Metaphor in NLP*, pages 40–49.

Ilana Heintz, Ryan Gabbard, Mahesh Srivastava, Dave Barner, Donald Black, Majorie Friedman, and Ralph Weischedel. 2013. Automatic extraction of linguistic metaphors with lda topic modeling. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 58–66.

Beata Beigman Klebanov, Ben Leong, Michael Heilman, and Michael Flor. 2014. Different texts, same metaphors: Unigrams and beyond. In *Proceedings of the Second Workshop on Metaphor in NLP*, pages 11–17.

Beata Beigman Klebanov, Chee Wee Leong, and Michael Flor. 2015. Supervised word-level metaphor detection: Experiments with concreteness and reweighting of examples. In *Proceedings of the Third Workshop on Metaphor in NLP*, pages 11–20.

Beata Beigman Klebanov, Chee Wee Leong, E Dario Gutierrez, Ekaterina Shutova, and Michael Flor. 2016. Semantic classifications for detection of verb metaphors. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 101–106.

George Lakofi and Mark Johnson. 1980. Metaphors we live by. *Chicago, IL: University of.*

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Michael Mohler, David Bracewell, Marc Tomlinson, and David Hinote. 2013. Semantic signatures for example-based linguistic metaphor detection. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 27–35.

Philippe Muller, Cécile Fabre, and Clémentine Adam. 2014. Predicting the relevance of distributional semantic similarity with contextual information. In *52nd Annual Meeting of the Association for Computational Linguistics-ACL 2014*, pages 479–488.

Raúl Santos-Rodríguez, Darío García-García, and Jesús Cid-Sueiro. 2009. Cost-sensitive classification based on bregman divergences for medical diagnosis. In *Machine Learning and Applications, 2009. ICMLA'09. International Conference on*, pages 551–556. IEEE.

Gerard J Steen, Aletta G Dorst, J Berenike Herrmann, Anna A Kaal, and Tina Krennmayr. 2010. Metaphor in usage. *Cognitive Linguistics*, 21(4):765–796.

Yulia Tsvetkov, Leonid Boytsov, Anatole Gershman, Eric Nyberg, and Chris Dyer. 2014. Metaphor detection with cross-lingual model transfer. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 248–258.

Peter D Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 680–690. Association for Computational Linguistics.

Chuhan Wu, Fangzhao Wu, Yongfeng Huang, Sixing Wu, and Zhigang Yuan. 2017. Thu_ngn at ijcnlp-2017 task 2: Dimensional sentiment analysis for chinese phrases with deep lstm. *Proceedings of the IJCNLP 2017, Shared Tasks*, pages 47–52.

Xuesong Yang, Anastassia Loukina, and Keelan Evanini. 2014. Machine learning approaches to improving pronunciation error detection on an imbalanced corpus. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*, pages 300–305. IEEE.