Chinese Relation Classification using Long Short Term Memory Networks

Linrui Zhang, Dan Moldovan
The University of Texas at Dallas
800 West Campbell Road ; MS EC31, Richardson, TX 75080 U.S.A
linrui.zhang@utdallas.edu, moldovan@hlt.utdallas.edu

Abstract

Relation classification is the task to predict semantic relations between pairs of entities in a given text. In this paper, a novel Long Short Term Memory Network (LSTM)-based approach is proposed to extract relations between entities in Chinese text. The shortest dependency path (SDP) between two entities, together with the various selected features in the path, are first extracted, and then used as input of an LSTM model to predict the relation between them. The performance of the system was evaluated on the ACE 2005 Multilingual Training Corpus (Walker et al., 2006), and achieved a state-of-the-art F-measure of 87.87% on six general type relations and 83.40% on eighteen subtype relations in this corpus.

Keywords: Long Short Term Memory Networks, Shortest Dependency Path, Chinese Relation Classification

1. Introduction

The task of relation classification is to predict semantic relations between pairs of entities. Formally, the goal is to predict the semantic relations between a head entity e_h and a tail entity e_t from a given sentence. For example, in the phrase "俄罗斯总统 Russian President", the relations between "Russia (ORG)" and "President (PER)" are ORG-AFF which indicate that the "President" is affiliated to an organization "Russia" , in ACE annotation ORG-AFF(Russia, President), and to be more specific, "President" is the employee of "Russia", EMPLOYMENT(Russia, President). In ACE corpus 2005, six general relations such as ORG-AFF and PART-WHOLE, and eighteen subtype relations such as OWNERSHIP, LOCATED, and EMPLOYMENT are defined.

The feature-based, kernel-based and deep learning-based methods are the most popular models for relation extraction in the literature. The basic idea of the feature-based approach is to treat relation extraction as a classification problem. Different kinds of features are extracted from text and then fed into a classifier. Such work includes (Kambhatla, 2004), (GuoDong et al., 2005) and (Moldovan and Blanco, 2012). However, these approaches suffer from error propagation problems. If the features are not well selected, the errors will be added up until the end. (Jiang and Zhai, 2007) systematically analyzed the effectiveness of different features for relation extraction on a large feature space and concluded that just using the basic unit features from each feature space (sequence, syntactic and dependency relation) can achieve reasonably good performance, and adding more complex features may not benefit the result. The kernel-based approach is to compute a kernel function to measure the similarity between two data objects. Such work includes (Zelenko et al., 2003), (Culotta and Sorensen, 2004), (Bunescu and Mooney, 2005), (Zhang et al., 2006), (Zhou et al., 2007), (Wang, 2008) and (Plank and Moschitti, 2013). The key issue of the kernel-based approach is the slow training and prediction time, so it is not good enough to process big data. With the development of deep learning, a series of neural network-based models are proposed, such as recursive neural network-based approaches (Socher et al., 2012),(Ebrahimi and Dou, 2015), and convolutional neural network-based approaches (Zeng et al., 2014), (Santos et al., 2015), and (Nguyen and Grishman, 2015).

Long short-term memory (Hochreiter and Schmidhuber, 1997) can capture long-term dependencies in sequences, so they could be used to model sequential data naturally. Recently they have been used frequently in many NLP tasks (Cho et al., 2014). Shortest dependency paths (SDP) have proven to be highly useful to relation extraction (Ebrahimi and Dou, 2015). They can to the utmost avoid involving irrelevant words in the path from one entity to another. (Xu et al., 2015) combined LSTMs and SDPs together and proposed a SDP-LSTM model for an English relation classification task on the SemEval-2012 dataset. Intrigued by this idea, we built a simplified SDP-LSTM model for Chinese relation classification on the ACE 2005 corpus and obtained state-of-the-art results. To the best of our knowledge, we are the first to implement an LSTM network for Chinese relation classification. This paper is organized as follows: We first present related work, then describe the system in detail, including the SDP and features for Chinese, and explain how to apply them to the LSTM Model. After that we present the details of our experiment, such as the corpus, the training details and the tools used. Finally, we compare our work with previous work in the literature, and show the effectiveness of different features on the final results.

2. Related Work

Research on Chinese relation extraction is quite limited compared with the progress with English. This may be due to two reasons. First, the Chinese language makes less use of function words and morphology (Levy and Manning, 2003), which makes it harder to extract syntactic information from it. Second, the lack of relevant corpora and tools also slows down the progress of research in Chinese. (Che et al., 2005) (Kebin et al., 2007) (Huang et al., 2008) (Yu et al., 2010), and (Dandan et al., 2012) proposed different kernel based approach to Chinese relation extraction. (Li et al., 2008) (Zhang et al., 2008),(Zhang et al., 2011) published a series of papers on feature-based approaches. In particular, they designed nine positional structures of enti-

ties and focused on the effectiveness of the positional feature on Chinese relation extraction. (Chen et al., 2014) proposed a novel Omni-word feature which takes advantage of Chinese sub-phrases, together with soft constraints for Chinese relation extraction, and got a 90.35% F1 score on Type relation, and a 75.44% F1 score on subType relation. Other interesting work such as (Chen et al., 2012) introduces a Deep Belief Network model that can handle high-dimensional feature space (Lin et al., 2010) worked on a mixed model that combined feature-based and kernel-based models together.

Word vector representation is the foundation of deep learning techniques for NLP. There are two popular models for word embedding: the word2vec model, which is promoted by Google (Mikolov et al., 2013), and the GloVe model, which is promoted by Stanford University (Pennington et al., 2014). Low-dimensional, dense word embeddings can effectively alleviate sparsity by sharing statistical strength between similar words, and can provide a good starting point to construct features of words and their interactions (Chen and Manning, 2014).

Long short term memory (LSTM) networks were first proposed by Hochreiter in 1997 (Hochreiter and Schmidhuber, 1997). An LSTM could be viewed as a complex activation unit that has an input gate, a forget gate, an output gate, a new memory cell and a final memory cell. Figure 1 shows the complete representation of a long short term memory unit.[1]

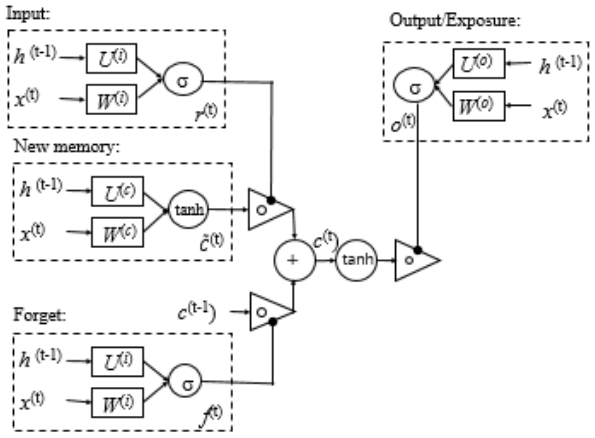

Figure 1: The complete representation of a long short term memory unit.

The mathematical formulation of LSTM units is as follows:

The input gate is to decide if the input x_t is worth being preserved based on the input word x_t and the past hidden state h_{t-1}.

$i_t = \sigma(W^{(i)}x_t + U^{(i)}h_{t-1})$

The forget gate f_t makes an assessment on whether the past memory cell is useful to compute the current memory cell.

$f_t = \sigma(W^{(f)}x_t + U^{(f)}h_{t-1})$

The output gate is to separate the final memory c_t from the hidden state.

$o_t = \sigma(W^{(o)}x_t + U^{(o)}h_{t-1})$

The new memory generation cell is used to generate a new memory $\tilde{c}_t$ by input work x_t and the past hidden state h_{t-1}.

$\tilde{c}_t = tanh(W^{(c)}x_t + U^{(c)}h_{t-1})$

The final memory cell produces the final memory c_t by summing the advice of the forget gate f_t and input gate i_t

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

$h_t = o_t \circ tanh(c_t)$

3. Approach

The pipeline of the system is as follows: we first find the shortest dependency path (SDP) between two given entities in a sentence. Second, we extract the selected features along the words in the SDP. Finally, we concatenate the distributated representation of the words and their features into dense vectors, and feed them as input of LSTM models.

3.1. The Shortest Dependency Path

Dependency parsing captures the dependence relations between words, and when compared with constituency paths, dependency paths have a better ability to encode information. In dependency parsing, the dependency relations and words will form a dependency graph. The edges are the dependency relations and the vertices are the words. Finding the shortest dependency path between words may be mapped into finding the shortest path between two vertices in the dependency graph. We used Stanford CoreNLP (Manning et al., 2014) for dependency parsing, and NetworkX (Hagberg et al., 2008) to get the shortest path in the dependency graph. Figure 2 and 3 show an example to map from the dependency graph to shortest dependency path of sentence "We went home after comforting her sister (我们在安慰完妹妹后也回家了).".

3.2. Feature selection

We extracted four kinds of features: word embedding, Part-of-Speech tags, entity type, and entity subtype.

We use Google's word2vec model and trained our word vectors on Chinese Wikipedia data. Part-of-Speech tags are extracted by Stanford CoreNLP. The type and subtype of the entities are already annotated by the corpus. The ACE corpus defines seven general types of entities, and each general type could be subcategorized into subtypes. There are 44 total subtypes of entities.

All the features use distributed representations and are only applied to the words that are on the shortest dependency path. For each word, we concatenated it with all its features into a dense vector and fed it as the input of the LSTM model. Figure 4 shows the feature representation of the sentence.

3.3. Model Structure

In general, the model has three layers: one LSTM layer, one dropout layer and one softmax layer. An overview of the model is shown in Figure 5.

3.3.1. The LSTM layer

The LSTM network takes input from the data directly. The input of the LSTM network should be in three dimensions.

[1] http://web.stanford.edu/class/cs224n/lecture_notes/cs224n-2017-notes5.pdf

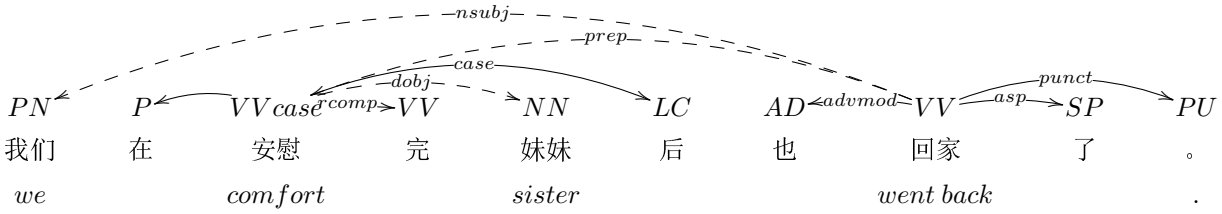

Figure 2: The dependency graph of the sentence "We went back after comforting her sister."

$$we \xleftarrow{nsubj} went\ back \xrightarrow{prep} comfort \xrightarrow{dobj} sister$$

Figure 3: The shortest dependency path between "we "and "sister ".

Word		POS		Type		SubType
We		PN		PER		$Group$
go	+	VV	+	$\emptyset$	+	$\emptyset$
$comfort$		VV		$\emptyset$		$\emptyset$
$sister$		NN		PER		$Individual$

Figure 4: feature representation of the sentence.

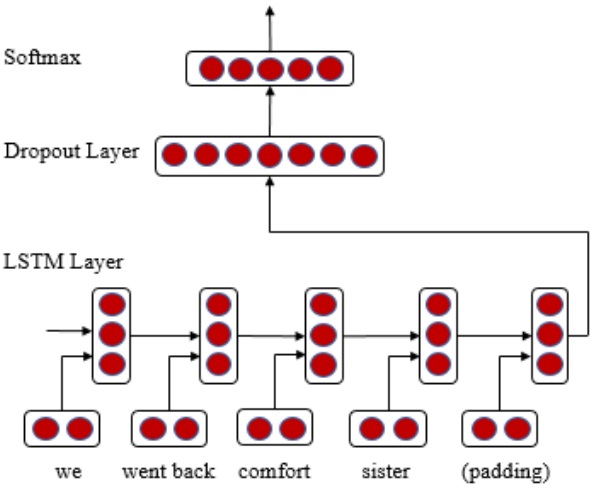

Figure 5: The strcture of our model.

The first dimension is the batch size, which is set during the experiment. The second dimension is the size of the input, which is the size of the dense vector that concatenates the word and its features. The third dimension is the time step dimension of the LSTM, which in our case is the sequence length of the Shortest Dependency Path (SDP). However, different instances have the different lengths of SDPs, so we use padding techniques that unify the SDP length of all the training examples.

3.3.2. The Dropout layer

A dropout layer is built on top of LSTM layer. It is used to reduce overfitting by randomly delete features from network during the training in order to prevent the features from excessively co-adaptation (Hinton et al., 2012) (Srivastava et al., 2014). To be more formal, a kept_prob rate of probability P is initialized. During the training process, 1- P of the features will be randomly deleted, however during the test process we do not delete any features, as a result, the testing process will have the same magnitude as the training process, and the overfitting will be prevented.

3.3.3. The softmax layer

A softmax layer is usually used as the output layer for classification problems in Deep Learning. It takes the output of the dropout layer and outputs the probability distribution of the candidate classes. In our case, the outputs are the six general type relations and the eighteen subtype relations between the entities.

3.3.4. Learning

The training goal is to minimize the cross-entropy error between the softmax layer outputs (the probabilistic distribution of the predicated relations) and the one-hot representation of the gold annotated relations. The mathematic representation of the training process is as follows:

$$E(\theta) = -\sum_{n}\sum_{k} t_n^k \log y_n^k + \frac{\lambda}{2}||\theta||^2$$

t is the gold annotated relations. y is the predicated relations from the softmax layer. λ is the regularization rate. θ is the model parameters we are trying to learn.

4. Experiments and Results

The model is implemented using Google's Tensorflow(Abadi et al., 2016). All the layers and the training object used the default packages included in Tensorflow. We did not make any changes to these packages.

4.1. Corpus and experiment details

The performance of the system was tested on the ACE 2005 training corpus Chinese data set. There are in total positive 7985 instances, and we split the corpus into 80% training and 20% testing.

We used the pretrained word vector on Chinese Wikipedia data and set the vector length to 60. Since there is no model proposed to train the Part-of-Speech tags and entity types, and they are not sparse even with one-hot representation,

Systems	features	Type F1	SubType F1
(Li et al., 2008)	Entity Type, Entity SubType 9 Position Strucute, N-grams Wordlist	70.29%	67.41%
(Zhang et al., 2011)	Entity Type, Entity SubType 9 Position Strucute, N-grams	70.43%	67.86%
(Dandan et al., 2012)	Convolution Tree Kernel	69.00%	66.60%
Ours	Entity Type, Entity SubType Word Embedding, Part-Of-Speech	**87.87%**	**83.40%**

Table 1: Performance of Relation Extraction Systems

we randomly initialized them and set the vector length to 20 for each of them, so the input size of the LSTM model is 120 dimensions. Because most relations in the ACE corpus are short-range relations, and some of them even within single noun phrases, we set the time step size to 5, and used padding to unify the sequence length. We also set the number of hidden units in the LSTM layer to 64 and kept the prob_rate on the dropout layer at 0.5.

4.2. Results

Table 2 shows the final results and effectiveness of different features to the result. From Table 2, we can conclude that all the selected features are effective and will benefit the results, and among those, word embedding is the most important feature. There is another popular feature, "positional structure" that is frequently used in previous research, and we believe our results will be further improved if we include it. This feature has been used particularly with the ACE corpus. However, modern corpus is no longer annotated with this feature, which is why we do not include it.

Systems	features	Type F1	SubType F1
LSTM	Word Embeddings	78.55%	72.92%
	+ POS	+1.63%	+1.67%
	+ Entity Type	+6.38%	+9.23%
	+ Entity SubType	+5.22%	+7.82%
	Overall	87.87%	83.40%

Table 2: The experiment results

4.3. Compare with state-of-the-art System

We compared our results with those of three previous approaches that were evaluated on the same corpus. Table 1 shows the comparison of the features/kernel use and the F1 scores of type and subtype relations between our approach and that of others. Previous works on this corpus focused on relation extraction problem. The different between relation extraction and relation classification is that relation extraction needs to detect if a relation utters corresponding to some entity pairs before predicting the relation between them. In practise, we could reduce the relation extraction problem into relation classification problem by removing negative instances (that do not have relations between entities). (Nguyen and Grishman, 2015) discovered that the system performace will improve about 20% from relation extraction to relation classification in ACE corpus. Considering this discovery, our system is still superior to the others. More importantly, our model does not combine tremendous feature engineering work, and this re-confirms its advantages.

4.4. Error analysis

Errors most frequently happen at the feature selection step. Stanford CoreNLP may make mistakes analyzing Chinese sentences. As with the entire pipeline model, it has error propagation issues.

Optimizing the feature embedding is another way to improve the results. For example, when we face a complex noun phrase that is not in the pretrained word vector, we may split it into several small elements, and then take the average of each element embedding. It is a simple strategy, but not the best. Besides, we randomly initialized the embeddings for the rest of features, however, as stated in (Chen and Manning, 2014), similar POS such as "NNS" and "NN", "VB" and "VBZ" should be clustered together and have similar embeddings, and it is also the same for entity type and subtype features.

5. Conclusion and Future Work

In this paper, we present an LSTM network with shortest dependency path model to extract relations between entities in Chinese. The results show that the model is effective on the ACE 2005 training corpus. Compared with traditional feature-based and kernel-based methods, deep learning models (neural network based methods) can easily achieve better results with fewer features.

There are several avenues for future work. The correct representation of the feature embedding is very important in deep learning models, and since Chinese is a character based language, some researchers propose that instead of using word embedding, we may use character embedding instead. In this case, we will no longer have segmentation problems, and have less feature engineering work. Since few deep learning based works have been done to Chinese, we would like to try more deep learning models and analyze their performance. Besides, our model is not very language-sensitive, so we will extend our work to other languages.

6. Bibliographical References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Bunescu, R. C. and Mooney, R. J. (2005). A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 724–731. Association for Computational Linguistics.

Che, W., Jiang, J., Su, Z., Pan, Y., and Liu, T. (2005). Improved-edit-distance kernel for chinese relation extraction. In *Proceedings of IJCNLP*, pages 132–137.

Chen, D. and Manning, C. D. (2014). A fast and accurate dependency parser using neural networks. In *Emnlp*, pages 740–750.

Chen, Y., Zheng, D.-Q., and Zhao, T.-J. (2012). Chinese relation extraction based on deep belief nets. *Ruanjian Xuebao/Journal of Software*, 23(10):2572–2585.

Chen, Y., Zheng, Q., and Zhang, W. (2014). Omni-word feature and soft constraint for chinese relation extraction. In *ACL (1)*, pages 572–581.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Culotta, A. and Sorensen, J. (2004). Dependency tree kernels for relation extraction. In *Proceedings of the 42nd annual meeting on association for computational linguistics*, page 423. Association for Computational Linguistics.

Dandan, L., Yanan, H., and Longhua, Q. (2012). Exploiting lexical semantic resource for tree kernel-based chinese relation extraction. *Natural Language Processing and Chinese Computing*, pages 213–224.

Ebrahimi, J. and Dou, D. (2015). Chain based rnn for relation classification. In *HLT-NAACL*, pages 1244–1249.

GuoDong, Z., Jian, S., Jie, Z., and Min, Z. (2005). Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 427–434. Association for Computational Linguistics.

Hagberg, A., Swart, P., and S Chult, D. (2008). Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Laboratory (LANL).

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Huang, R., Sun, L., and Feng, Y. (2008). Study of kernel-based methods for chinese relation extraction. *Information Retrieval Technology*, pages 598–604.

Jiang, J. and Zhai, C. (2007). A systematic exploration of the feature space for relation extraction. In *HLT-NAACL*, pages 113–120.

Kambhatla, N. (2004). Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 22. Association for Computational Linguistics.

Kebin, L., Fang, L., Lei, L., and Ying, H. (2007). Implementation of a kernel-based chinese relation extraction system [j]. *Journal of Computer Research and Development*, 44(8):1406–1411.

Levy, R. and Manning, C. (2003). Is it harder to parse chinese, or the chinese treebank? In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 439–446. Association for Computational Linguistics.

Li, W., Zhang, P., Wei, F., Hou, Y., and Lu, Q. (2008). A novel feature-based approach to chinese entity relation extraction. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 89–92. Association for Computational Linguistics.

Lin, R., Chen, J., Yang, X., and Xu, H. (2010). Research on mixed model-based chinese relation extraction. In *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, volume 1, pages 687–691. IEEE.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., and McClosky, D. (2014). The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Moldovan, D. I. and Blanco, E. (2012). Polaris: Lymba's semantic parser. In *LREC*, pages 66–72.

Nguyen, T. H. and Grishman, R. (2015). Relation extraction: Perspective from convolutional neural networks. In *VS@ HLT-NAACL*, pages 39–48.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.

Plank, B. and Moschitti, A. (2013). Embedding semantic similarity in tree kernels for domain adaptation of relation extraction. In *ACL (1)*, pages 1498–1507.

Santos, C. N. d., Xiang, B., and Zhou, B. (2015). Classifying relations by ranking with convolutional neural networks. *arXiv preprint arXiv:1504.06580*.

Socher, R., Huval, B., Manning, C. D., and Ng, A. Y. (2012). Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1201–1211. Association for Computational Linguistics.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way

to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Walker, C., Strassel, S., Medero, J., and Maeda, K. (2006). Ace 2005 multilingual training corpus. *Linguistic Data Consortium, Philadelphia*, 57.

Wang, M. (2008). A re-examination of dependency path kernels for relation extraction. In *IJCNLP*, pages 841–846.

Xu, Y., Mou, L., Li, G., Chen, Y., Peng, H., and Jin, Z. (2015). Classifying relations via long short term memory networks along shortest dependency paths. In *EMNLP*, pages 1785–1794.

Yu, H., Qian, L., Zhou, G., and Zhu, Q. (2010). Chinese semantic relation extraction based on unified syntactic and entity semantic tree. *Journal of Chinese Information Processing*, 24(5):17–23.

Zelenko, D., Aone, C., and Richardella, A. (2003). Kernel methods for relation extraction. *Journal of machine learning research*, 3(Feb):1083–1106.

Zeng, D., Liu, K., Lai, S., Zhou, G., Zhao, J., et al. (2014). Relation classification via convolutional deep neural network. In *COLING*, pages 2335–2344.

Zhang, M., Zhang, J., Su, J., and Zhou, G. (2006). A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 825–832. Association for Computational Linguistics.

Zhang, P., Li, W., Wei, F., Lu, Q., and Hou, Y. (2008). Exploiting the role of position feature in chinese relation extraction. In *LREC*.

Zhang, P., Li, W., Hou, Y., and Song, D. (2011). Developing position structure-based framework for chinese entity relation extraction. *ACM Transactions on Asian Language Information Processing (TALIP)*, 10(3):14.

Zhou, G., Zhang, M., Ji, D.-H., and Zhu, Q. (2007). Tree kernel-based relation extraction with context-sensitive structured parse tree information. In *EMNLP-CoNLL*, volume 2007, pages 728–736.