2017 13th International Conference on Intelligent Environments

Relation Extraction via Position-Enhanced Convolutional Neural Network

Weiwei Shi
Beijing University of Posts and Telecommunications
Beijing, China
Email: sww9370@bupt.edu.cn

Sheng Gao
Beijing University of Posts and Telecommunications
Beijing, China
Email: gaosheng@bupt.edu.cn

***Abstract*—Recently, deep neural network based methods have been widely used in relation extraction, which is an important task for knowledge base population, question answering and other natural language applications, to learn proper features from entities pairs and other sentence parts to extract relations from text. As a kind of important information, the value of position is always been underestimated, which causes a low weight of position information in various models and finally hurts the performance of relation extraction task. To alleviate this issue, we propose a position-enhanced embedding model based on convolutional neural network. In this model, we split the sentence representation into three parts based on the entity pairs in the sentence, and use three independent convolutional networks to learn features. Furthermore, we concatenate the output from different branches and employ a softmax layer to compute the probability for each relation. Experimental results on wildly used datasets achieve considerable improvements on relation extraction as compared with baselines, which shows that our proposed model can make full use of position information.**

***Keywords*-convolutional neural network; relation extraction; position information**

I. Introduction

There are many attempts to solve relation extraction problem in the last decades, which can be divided into two main approaches as feature-based methods and kernel-based methods. Feature-based methods generally first selected a set of proper features by different analysis from text, such as part-of-speech(POS) tagging to full parsing and dependency parsing, etc[1][2][3][4]. and then used some statistical classifier such as Support Vector Machines (SVM) or Maximum Entropy (MaxEnt) to classify the unlabeled entity pairs. Different from feature-based methods, the main idea of kernel-based methods is to design kernel functions on some structure representation such as sequences or trees of the relation instances to capture similarity between two relation instances[5][6]. These methods still suffer from the difficulty of selecting proper features manually and errors generate by NLP tools inevitably.

With the tremendous successes achieved by deep neural networks on natural language processing[7], many researchers employed neural-based model trying to reduce the reliance of NLP tools and human efforts. Socher[8] presented a recursive neural network that learns compositional vector representations for phrases and sentences by assigning a vector and a matrix to every node in a parse tree, and Hashimoto[9] proposed a recursive neural network based on a static tree. Although these neural-based models above achieved considerable results, but they still relied on NLP tools, to further reduce the reliance of NLP tools, Zeng[10] proposed a method that used lexical-feature (wordnet and n-gram feature of entity pairs) and sentence-level feature (word-embedding and position-embedding) as the input of convolutional neural network, and used this network to extract the sentence feature for classification. Furthermore, some convolutional methods only adopted sentence-level features as the input of neural network and achieve better performance than other traditional feature based method[11][12][13]. Though these methods used position embedding to take position effects into account, they still underestimated the importance of position information, which causes a low weight of position information in their model and finally hurts the performance. Since position features reflects the structure features in some way, it can help to learn the latent information for relation extraction[10], which is important for us to make the best use of position information, thus, we proposed a novel model which can enhance the position information in convolution framework.

To be concrete, we assume that different parts of sentence reveals different structure feature, and performance would be weaken if we merge all the sentence representation in convolution and pooling process, so, we split the sentence embedding according to the position of entities and learn different part of sentence to gain the local feature, after that, we concatenate all local features to generate the global feature of sentence for classification.

The contributions of this paper can be summarized as follows:

1) We explore the feasibility of enhancing the position information by learning different part of sentences. A position-enhanced embedding convolutional network is employed to extracting sentence feature while reserving

2472-7571/17 $31.00 © 2017 IEEE
DOI 10.1109/IE.2017.28

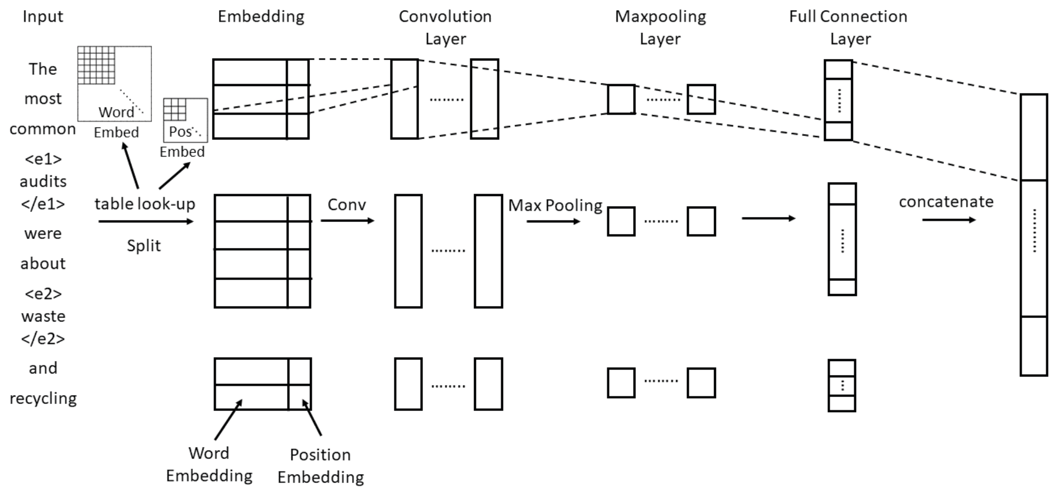

Figure 1. Framework of Position enhancement CNN network

position information.

2) We conduct experiments using the ACE2005 and SemEval-2010. Results of experiments reveals that proposed position enhanced method are effective in both relation extraction and relation classification task. Our method outperform the baseline methods.

II. Related work

This paper extracts binary relation of the form (entity_1, relation, entity_2). The most widely used methods for this task are supervised methods. In supervised methods, relation extraction is treated as multi-class classification problem, thus, a sentence classifier is trained to distinguish features from different relation mentions. As described in last section, feature-based and kernel-based method are two main kind of traditional supervised method, but the performance of these two methods is largely influenced by the quality of supervised NLP tools and manual features.

Recent years, neural networks has been widely used in relation extraction area, researchers put a lot of effort into investigating ways to reduce the reliance on human labor and NLP tools by neural network. Socher[8] introduced a recursive neural network model that learns compositional vector representation for phrases and sentences of arbitrary type and length. Since Socher's model is based on constituency-based parsers which may span words that do not contribute to the relation, Javid[14] alleviated this issue by introducing a compositional account of dependency graphs that can match RNNs recursive nature. RNN-based framework also used in Zhang[15] and Xu[16]'s work. As for convolutional network, Thien[17] proposed a novel CNN framework which takes advantages of multiple window sizes for filter and pre-training word embedding, in addition, inputs of Thiens model didn't require additional NLP tool and manual features. After that, Zeng[18] optimize Thien's model by introduce a dynamic pooling layer and Dos[19] proposed a new pairwise ranking loss function that replace the softmax layer to gain better performance. In addition, there are methods combines two neural network[12][20]. Although methods above achieve solid results, ideally, we would want to further improve its performance by enhancing position information without the help of NLP tools. Our experiments on Section 4 reveals that better performance would be achieved by applying the position-enhanced method.

III. Model

To make full use of position information and gain better performance in relation extraction and relation classification task, we propose a position-enhanced embedding model, details of this model will be introduced in this section.

A. Framework for position-enhanced embedding model

Framework of our proposed model is shown in Fig.1, it (i) encodes an input sentence by real-valued vectors, (ii) then splits the feature matrix into three parts according to the position of two entities, (iii) after that, employs three independent convolutional neural networks to learn the local features from different part of the sentence, (iv) and finally, concatenates the outputs into a single vector as the sentence global feature and uses a softmax layer to obtain the conditional probability matrix which describes the probability of different relation. Each network describe in (iii) contains a convolution layer, a max-pooling layer and a full-connection layer as shown in Fig.1.

B. Sentence-level representation

Inputs of our model are raw word tokens. The first step of preprocess is to turn these word tokens into low-dimensional vectors, we adopt the same preprocess method as Zeng[4] do in his task.

Rule	**Example**
Cut Head:	~~The … not~~ just laying India's … family of
Cut Middle:	<e1>Contractions**</e1>…dia**phragm are ~~caused … nerve~~ either by the brain .. <e2>irritation</e2>
Cut Tail:	and advice on current United States ~~government … issues.~~

Figure 2. Trimming Rule

Suppose a sentences length is n, and let $s = [t_1, ..., t_{n1}, ..., t_{n2}, ..., t_n]$, t_i is the i-th token in this sentence, t_{n1} and t_{n2} are two entity tokens. We firstly transform tokens into word embedding vector e_i by looking up the existing pre-trained word embedding table V. Then, we compute the relative distances between tokens and each entity. Suppose the range of relative distance is -n to n, we generate two random matrix PF1 and PF2 with the dimension of $(2n+1)m_d$ as the position embedding tables(m_d is the dimension of position embedding, we will discuss its effect in Experiment part). After that, we map the relative distances into embedding by looking up the position embedding table PF1 and PF2, and each token can be represented as $t_i = [e_i, d_{i1}, d_{i2}]$, here, d_{i1} is vector from PF1 and d_{i2} is vector from PF2. At last, we concatenate token features and gain a 2-dimension matrix $(m_e + 2m_d)n$ for each sentence.

C. Feature Segmentation and Trimming

Position features are widely used in convolution-based model, but it suffers noise come from convolution and pooling process which hurt the final process, to better reserve position information and further enhance it, we divide s into three parts according to the position of two entities, here, $s_1 = [t_1, .., t_{n1-1}]$, $s_2 = [t_{n1}, .., t_{n2}]$, $s_3 = [t_{n2+1}, .., t_n]$, t_i is the feature representation of i-th word in sentence as described in last section.

Since the input dimension of CNN is fixed, we should trim or pad sentence before training. We first calculate the length of each part in our dataset and choose the length which is longer than 80% of sentences as the input length. In addition, to better reserve the information of long sentence, we trim the words which are far away from entities as shown in Fig.2. After trimming, we gain a sentence representation $s = [s_1, s_2, s_3]$, and put s_i into i-th branch of our net.

D. Feature Convolution Component

The aim of convolution layer is to extract higher level features, as shown in Fig.1, there are three branches of convolution, we expect each branch can learn different local feature from different part. The setting (window size, step length, learning rate and so on) of each branch are the same.

The input of convolution layer is sentence embedding represented by attaching word presentations, which could only represent the feature of word, since we can't predict relation by words, thus, we use the convolution layer to merge the word feature and generate the sentence feature. Suppose m_w to be the convolution window size, and a convolution filter can be treat as a weight matrix $W = [w_1, w_2, , w_{m_w}]$, where $w_i \in R^{m_w m_e + 2m_d}$. Given the input $s_i = [t_1, .., t_n]$, we gain the convolutional output $O_x^1 = [o_1, o_2, ..., o_{nl-m_w+1}]$, here $x \in 1, 2, 3$ represents output from different branchs convolution layer, o_k can be computed as follows:

$$o_k = \delta(\sum_{j=0}^{m_w} w_{j+1}^T t_{j+k}^T + b) \quad (1)$$

Where b is a bias term and δ is a non-liner function. As shown in (1), we can see that as the window moves, convolutional filter learns n-gram feature from sentences, and we can use various filters to learn different kind of feature from sentences.

After convolution, pooling layer is to choose the most useful feature in the output of convolutional layer while reduce the amount of parameters. Given the convolution layer output $O_x^1 = [o_1, o_2, ..., o_{nl-m_w+1}]$, the out of pooling layer is computed as follows:

$$p_x = \max(o_1, o_2, , o_{nl-m_w+1}) \quad (2)$$

E. Relation inference component

Full connection layer comes after max-pooling layer, at this step, the pooling output for every convolution filter are concatenated into a 1-dimension vector $P = [p_1, p_2, ..., p_f]$ to represent the input sentence, here f is the amount of convolution filters. The full connection layer output is compute as follows:

$$O_x^2 = C_x P_x^T \quad (3)$$

Where $C_x \in R^{m_c f}$, (m_c is numbers of neurons.) can be seen as the weight matrix of full connection layer, as shown in Fig.1, each convolution branch extract local features from part of sentences, and we can't predict a relation precisely by any part of them, so we concatenate the features together and use it to classify.

At last, we use a softmax layer to compute the conditional probability for each relation as follows:

$$p(r|S, \theta) = \frac{\exp(O_r^3)}{\sum_{k=1}^{n_r} \exp(O_k^3)} \quad (4)$$

Where n_r the total number of relations, θ indicates our model parameters and O^3 is the final output of the neural network which defined as follows:

$$O^3 = MO^2 + d \quad (5)$$

Where $O^2 = [O_1^2, O_2^2, O_3^2]$ is the feature which concatenated by the output of each branchs full connection layer, d is a bias term and M is the representation matrix of relations.

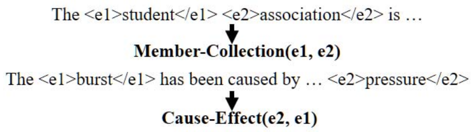

Figure 3. directional information in SemEval dataset

F. Model learning

The objective function of our model is shown as follows:

$$J(\theta) = \sum_{i=1}^{m_s} \log p(r_i|S_i, \theta) \tag{6}$$

Where m_s is the total number of training sentences. We adopt stochastic gradient descent (SGD) with shuffled mini-batches and the AdaDelta update rule[21] to solve this optimization problem. Besides, to avoid overfitting, we use l2-regularzation to rescale the weights of full connection layer.

IV. Experiments

Experiments are conducted to demonstrate that the position embedding are critical to relation extraction and the enhancement of position information can greatly improve the performance of it. To this end, we firstly introduce the datasets and open resources used in our experiments, next, details of hyper-parameters and evaluation metrics will be discussed, then, we explore the effects of position embedding and convolutional filter size in our model, At last, to evaluate the effects of our position enhanced method, we compare our model with several state-of-art non-position-enhanced method on both relation extraction and relation classification task.

A. Datasets and open resources

Our experiments have been conducted on two datasets to evaluate the effectiveness of our Position-Enhanced-CNN model in relation classification and relation extraction task, including SemEval-2010 Task 8 dataset[22] and ACE2005 dataset[23]. SemEval-2010 Task 8 is for relation classification task and ACE2005 is for relation extraction task.

The SemEval data is available here[1]. There are 10,717 annotated instances, 8,000 for training and 2,717 for testing. As shown in Table I, there are 9 relations in this dataset, and each relation except Others has two directions. Since the annotation in SemEval dataset of e1 and e2 only represent its position order in the sentence as shown in Fig.3, the official evaluation method take directionality into account, which means we should judge the relation and its directionality for each instances, thus, 19 labels are possible.

The details of ACE2005 dataset is shown in Table II, there are 6 relations in this dataset, to generate the ”other” relations, we firstly collect sentences which contains two entities and didn’t included in the annotated relation set, then exclude the sentences generated by previous step whose distances between the two entity heads are greater than 15 to reduce the noise, as Thien[3] do in his job. After that, we gain an unbalanced dataset of 6,823 positive instances annotated by 6 relations and 65,846 negative examples for relation ”other”.

To achieve the better performance, we used the pre-trained word embedding from Mikolov[17] for English dataset. This embedding[2] are trained on 100 billion words of Google News using the continuous bag-of-words architecture, its

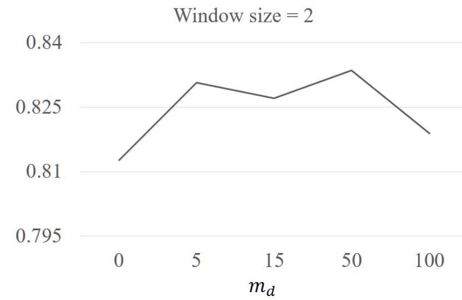

Figure 4. Results on different position size

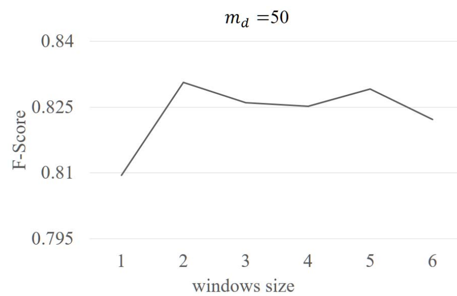

Figure 5. Results on different window size

Table I
Details of Semeval2010 dataset

Relation	%
Cause-Effect	12.4
Component-Whole	11.7
Entity-Destination	10.6
Entity-Origin	9.1
Product-Producer	8.8
Member-Collection	8.6
Message-Topic	8.4
Content-Container	6.8
Instrument-Agency	6.2
Other	17.4

[1]http://docs.google.com/View?id=dfvxd49s_36c28v9pmw

[2]https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit?pref=2&pli=1

Table II
DETAILS OF ACE2005 DATASET

Relation	%
ORG-AFF	2.6
PER-SOC	1.1
ART	0.9
PART-WHOLE	1.4
GEN-AFF	1
PHYS	2.1

Table III
DETAILS OF BASELINE MODEL

Model	Features used to train
Words	Words in the context before mention1
	Words after mention2 and between two mentions
	Bigrams, including word sequences between two entities, order of two mentions, number of words between two mentions
Words-WC-Web	Add the word embedding of context
Words-HM-Web	Add the word embedding of head words
T-CNN	Position feature, word embedding
Our CNN	Position feature, word embedding

Table IV
RESULTS OF RELATION EXTRACTION

System	P	R	F
Words	54.95	43.73	48.69
Words-WC-Web	50.10	44.47	47.11
Words-HM-Web	57.01	55.74	56.36
T-CNN	71.25	53.91	61.32
OurCNN	67.33	**73.36**	**70.21**

Table V
RESULTS OF RELATION CLASSIFICATION

Model	Features	F1
SVM	-POS, stemming, syntactic patterns	60.1
	-POS, stemming, syntactic patterns, WordNet	72.5
	-POS, prefixes, morphological, WordNet, dependency, Levin classed, Probank, FrameNet, NomLex-Plus, Google n-gram, paraphrases, TextRunner	82.2
MaxEnt	-POS, morphological, noun compound, thesauri, Google n-grams, WordNet	77.6
RNN	-	74.8
	-POS, NER, WordNet	77.6
O-CNN	- Lexical Level Feature	78.9
	- Lexical Level Feature, WordNet	82.7
T-CNN	-	2.8
Our-CNN	-	**83.1**

dimensionality $m_e = 300$. Words not included in this embedding are initialized randomly by using uniform method in the range -1 to 1.

B. Hyperparameters as evaluation metrics

For all the experiments below, we use: relu for the non-linear function, 300 filters for each branch of convolution network layer and 230 neurons for each full connection layer, and the input length for each branch is 15, 15, 20. Regarding other parameters, we use mini-batch size of 50, the hyperparameter for l2-regularzation of 0.5, and the learning rate is 0.1.

The official ranking of the participating systems on SemEval 2010 dataset is macro-averaged F1-score, we also adopt this evaluation metrics on SemEval 2010 experiments conduct by ourselves. As for ACE2005 dataset, we used the Precision, Recall and F1-score as Thien[11] do in his experiment.

C. Evaluation of position embedding and filter size

In this section, we explore the effects of position embedding and convolutional filter size by running our proposed position enhance CNN model.

Although position embedding are widely used in many relation extraction model, there still not tasks discuss its effectiveness and suitable dimension to our knowledge. We run this experiment on the SemEval dataset and set the dimension among $\{0, 5, 15, 50, 100\}$, the window size of filter is fixed to 2 and the evaluation matrix is F-score. Experimental result is shown in Fig.4. From this figure, we can observe that performance become better with the increase of position embedding dimension, but when the dimension increased to 100, performance become worse, assuming large proportion of random feature will add more noise and weak its performance. In this experiment, the most suitable position embedding dimension is 50.

Since different window size of filters will merge different range of input feature, choosing a suitable window size is vital to our model. In this experiment, we set the window size among $\{1, 2, 3, 4, 5, 6\}$, m_d is fixed to 50, evaluation matrix is F-score, SemEval dataset is used to evaluate. The experimental result is shown on Fig.5, we can see that the performance was weak when the window size set to 1, performance is not improved when the window size set to 6, and we achieve the best result when the window size set to 2. From this result, we can conclude that convolution based model can learn more information by merge features, but with the increase of merge range, weights of important feature might decrease, which hurt the performance

D. Relation extraction experiment

We treat relation extraction task as relation classification on an unbalanced dataset ACE2005.

To evaluate the effects of our position enhanced method, we compare our model with the CNN-based models proposed by Thien[11] which benefit from merging outputs of convolutional filters with different window size. The key difference between our model and Thiens model T-CNN is that we don't adopt multi-window size of convolutional filters which makes our framework simpler, in addition, we use three CNN branches to learn from different part of sentences to enhance the position information. Besides, we also introduce traditional feature based methods Words[1], Words-WC-Web[2], and Words-HM-Web[6] as our baseline model. As shown in Tabel III, these three model armed different features, classification method they adopted is Max-Ent. Performance of each model via 5-flod cross validation on ACE2005 dataset is shown in Table IV.

We can find that with the word embedding of contexts feature harms the performance of Words-WC-Web model, but the word embedding of head words feature boost the performance of Words-HM-Web model which demonstrates the importance of picking suitable representation on the feature based systems. Different from feature based system, Thien and our method only require the word embedding and position embedding as the model input, these representation derived from raw sentences and didn't need manual work. More importantly, we can see that both Thien and our method outperform all the feature based model, and our model have better recall and F1-score than Thien, which indicates that our proposed position enhance method is reasonable.

E. Relation classification experiment

To further evaluate the effectiveness of our proposed model, we test it on SemEval 2010 dataset which used to be the relation classification task. We choose six state-of-art model for competition, including traditional feature based model and recent neural based model. For traditional feature based model, we choose SVM[22] and MaxEnt[22], and for neural based model, recursive neural based model(RNN)[8] is chose, we also choose the convolutional based model which includes O-CNN[10] and T-CNN[11]. Details of baseline model and experimental result is shown in Table V.

From the Table V we can observed that our model achieve best performance without rich feature. The key observation from this table is that (i) traditional model and most of neural-based model would benefit much from various features as describe in previous section. (ii) And all neutral base models outperforms traditional feature-based model for more latent information can be learned in training process. (iii) Besides, CNN-based models gains better performance compared with RNN-based models, because CNN-based model learns n-gram feature from sentence by convolution process which is useful for relation extraction[11]. (iv) Both O-CNN and T-CNN didn't do the position enhancement task as we do, thus, their performance is worse than ours.

V. Conclusion

We propose a position enhancement convolutional architecture for relation extraction. This network uses three independent branches to learn different part of sentences, as for the sentence representation, we only used the pre-trained word embedding and randomly generated position embedding. Experimental results reveals that our proposed architecture is reasonable and effective for both relation extraction and relation classification task. Our future work includes: (i) enrich the representation of embedding by introducing the novel knowledge bases, and (ii) explore other training method such as adversarial network to improve the performance.

References

[1] Z. GuoDong, S. Jian, Z. Jie, and Z. Min, "Exploring various knowledge in relation extraction," in *Proceedings of the 43rd annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2005, pp. 427–434.

[2] T. H. Nguyen and R. Grishman, "Employing word representations and regularization for domain adaptation of relation extraction." in *ACL (2)*, 2014, pp. 68–74.

[3] N. Kambhatla, "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*. Association for Computational Linguistics, 2004, p. 22.

[4] E. Boschee, R. Weischedel, and A. Zamanian, "Automatic information extraction," in *Proceedings of the International Conference on Intelligence Analysis*, vol. 71. Citeseer, 2005.

[5] D. Zelenko, C. Aone, and A. Richardella, "Kernel methods for relation extraction," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1083–1106, 2003.

[6] R. J. Mooney and R. C. Bunescu, "Subsequence kernels for relation extraction," in *Advances in neural information processing systems*, 2006, pp. 171–178.

[7] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv preprint arXiv:1404.2188*, 2014.

[8] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "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*. Association for Computational Linguistics, 2012, pp. 1201–1211.

[9] K. Hashimoto, M. Miwa, Y. Tsuruoka, and T. Chikayama, "Simple customization of recursive neural networks for semantic relation classification." in *EMNLP*, 2013, pp. 1372–1376.

[10] D. Zeng, K. Liu, S. Lai, G. Zhou, J. Zhao *et al.*, "Relation classification via convolutional deep neural network." in *COLING*, 2014, pp. 2335–2344.

[11] T. H. Nguyen and R. Grishman, "Relation extraction: Perspective from convolutional neural networks," in *Proceedings of NAACL-HLT*, 2015, pp. 39–48.

[12] ——, "Combining neural networks and log-linear models to improve relation extraction," *arXiv preprint arXiv:1511.05926*, 2015.

[13] N. T. Vu, H. Adel, P. Gupta, and H. Schütze, "Combining recurrent and convolutional neural networks for relation classification," *arXiv preprint arXiv:1605.07333*, 2016.

[14] J. Ebrahimi and D. Dou, "Chain based rnn for relation classification." in *HLT-NAACL*, 2015, pp. 1244–1249.

[15] D. Zhang and D. Wang, "Relation classification via recurrent neural network," *arXiv preprint arXiv:1508.01006*, 2015.

[16] Y. Xu, L. Mou, G. Li, Y. Chen, H. Peng, and Z. Jin, "Classifying relations via long short term memory networks along shortest dependency paths." in *EMNLP*, 2015, pp. 1785–1794.

[17] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[18] D. Zeng, K. Liu, Y. Chen, and J. Zhao, "Distant supervision for relation extraction via piecewise convolutional neural networks." in *EMNLP*, 2015, pp. 1753–1762.

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

[20] Y. Liu, F. Wei, S. Li, H. Ji, M. Zhou, and H. Wang, "A dependency-based neural network for relation classification," *arXiv preprint arXiv:1507.04646*, 2015.

[21] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[22] I. Hendrickx, S. N. Kim, Z. Kozareva, P. Nakov, D. Ó Séaghdha, S. Padó, M. Pennacchiotti, L. Romano, and S. Szpakowicz, "Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals," in *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*. Association for Computational Linguistics, 2009, pp. 94–99.

[23] R. Grishman, D. Westbrook, and A. Meyers, "Nyus english ace 2005 system description," *ACE*, vol. 5, 2005.