Relation Classification Using Segment-Level Attention-based CNN and Dependency-based RNN

Van-Hien Tran[1], **Van-Thuy Phi**[1], **Hiroyuki Shindo**[1,2], and **Yuji Matsumoto**[1,2]

[1] Nara Institute of Science and Technology
[2]RIKEN Center for Advanced Intelligence Project (AIP)
[1]{tran.van_hien.ts1,phi.thuy.ph8,shindo,matsu}@is.naist.jp

Abstract

Recently, relation classification has gained much success by exploiting deep neural networks. In this paper, we propose a new model effectively combining Segment-level Attention-based Convolutional Neural Networks (SACNNs) and Dependency-based Recurrent Neural Networks (DepRNNs). While SACNNs allow the model to selectively focus on the important information segment from the raw sequence, DepRNNs help to handle the long-distance relations from the shortest dependency path of the related entities. Experiments on the SemEval-2010 Task 8 dataset show that our model is comparable to the state-of-the-art without using any external lexical features.

1 Introduction

Relation classification (RC) is a fundamental task in Natural Language Processing (NLP) that aims to identify semantic relations between pairs of marked entities in given sentences (instances). It has attracted much research effort as it plays a vital role in many NLP applications such as Information Extraction and Question Answering (Nguyen and Grishman, 2015). Traditional approaches (Kambhatla, 2004; Zhang et al., 2006) usually rely heavily on hand-crafted features and lexical resources, or elaborately designed kernels, which are time-consuming and challenging to adapt to novel domains. Recently, neural network (NN) models have dominated the work on RC since they can effectively learn meaningful hidden features without human intervention.

However, most previous NN models only exploit one of the following structures to represent relation instances: raw word sequences (Zhou et al., 2016; Wang et al., 2016) and dependency trees (Wen, 2017; Le et al., 2018). While raw sequences can provide all the information of relation instances, they also add noise to the models from redundant information. While dependency tree structures help the models focus on the concise information captured by the shortest dependency path (SDP) between two entities, they lose some supplementary context in the raw sequence. It is clear that the raw sequence and SDP highly complement each other. We, therefore, combine them to be more effective in determining the relation without losing any information.

While CNNs are able to learn short patterns (local features) (LeCun et al., 1995), RNNs have been effective in learning word sequence information (long-distance features) (Chung et al., 2014). In this paper, we present a new model combining both CNNs and RNNs, exploiting the information from both the raw sequence and the SDP.

Our contributions are summarized as follows:

(a) We combine Entity Tag Feature (ETF) (Qin et al., 2016) and Tree-based Position Feature (TPF) (Yang et al., 2016) to improve the semantic information between the marked entities in the raw input sentences.

(b) We propose Segment-Level Attention-based Convolutional Neural Networks (SACNNs) which automatically pay special attention to the important text segments from the raw sentence for RC.

(c) We build Dependency-based Recurrent Neural Networks (DepRNNs) on the SDP to gain long-distance features. Then, we combine the SACNN and the DepRNN to preserve the full relational information. Our proposed model achieves new state-of-the-art results on SemEval-2010 Task 8, compared with other complex models.

2 Related Work

RC plays a significant role in many NLP applications. Recent work usually present the task from a supervised perspective.

Traditional supervised approaches can be divided into feature-based methods and kernel methods. Feature-based methods focus on extracting and combining relevant features. Rink and Harabagiu (2010) leveraged useful features to achieve the best performance on SemEval-2010 Task 8. Meanwhile, kernel methods measure the structural similarity between two data samples, based on carefully designed kernels. Wang (2008) combined convolutional kernel and syntactic features to gain benefits for relation extraction.

Nowadays, deep neural networks are widely utilized in RC. Zeng et al. (2014) exploited a CNN to extract lexical and sentence features. Qin et al. (2016) used ETF to specify target entities in input sentences and fed them to a CNN. Vu et al. (2016) combined CNN and RNN to improve performance. Some recent work leveraged SDP for RC. Yang et al. (2016) proposed a position encoding CNN based on dependency parse trees, while Wen (2017) presented a model that learns representations from SDP, using both CNN and RNN.

3 Our Method

Given a sentence S with an annotated pair of entities ($e1$, $e2$), we aim to identify the semantic relation between them. Since the set of target relations is pre-defined, RC can be treated as a multi-class classification problem. In this section, we describe our model in detail for resolving this problem.

3.1 Input Representation

In Figure 1, Entity Tag Feature (ETF) is firstly used to annotate two entities in each raw sentence. Then, each word is represented by the concatenation of two parts: Word Embedding (WE) and Tree-based Position Features (TPFs). The representation sequence is then fed to the SACNN.

Entity Information. As the pairs of entities ($e1$, $e2$) are previously known, it is important to provide their information to the NNs. Following the work of Qin et al. (2016), we also use ETF which involves adding four tokens: $\langle e1S\rangle$, $\langle e1E\rangle$, $\langle e2S\rangle$ and $\langle e2E\rangle$ to each input sentence.

Word Embedding. Distributed representations of words in a vector space have helped learning algorithms to achieve better performance in NLP tasks (Mikolov et al., 2013). Following most previous work, we also use pre-trained word embeddings to initialize input word tokens in our model.

Tree-based Position Features. Yang et al. (2016) proposed TPFs for encoding relative distances of the current word to marked entities in dependency trees. The relative distance refers to the length of the SDP between the current word and the target entity. Then, each integer number is represented by a randomly initialized vector. Since TPFs help the neural network focus on crucial words and phrases in a sentence (Yang et al., 2016), we therefore utilize TPFs in our model. In Figure 1, TPF$_1$ and TPF$_2$ are relative distance features of each word to $e1$ and $e2$, respectively. For the four tokens: $\langle e1S\rangle$, $\langle e1E\rangle$, $\langle e2S\rangle$ and $\langle e2E\rangle$, which do not belong to the dependency tree, we simply pad zero vectors for their TPFs.

SDP. For the input of the DepRNN, we merely use the SDP between two marked entities from the original sentence as in Figure 1. Each normal word in the SDP is represented by a vector from pre-trained word embeddings. Meanwhile, following Le et al. (2018), we also consider dependency relations between words in the SDP and represent each dependency relation d_i as a vector D_i that is the concatenation of two vectors as follows:

$$D_i = Dtyp_i \oplus Ddir_i,$$

where *Dtyp* is the undirected dependency vector (*i.e.,* nmod), and *Ddir* is the orientation of the dependency vector (*i.e.,* left-to-right or vice versa). Both *Dtyp* and *Ddir* are initialized randomly.

3.2 Framework

The architecture of our model is illustrated in Figure 1. The example sentence with two entities $e1$ (*play*) and $e2$ (*religion*) is labeled by the directional relation "*Message-Topic(e1;e2)*". While the raw sequence is passed to the SACNN, the SDP between $e1$ and $e2$ is used in the DepRNN.

Segment Attention-based CNN. In the SACNN, each raw sentence is divided into three segments according to two entities: the left segment, the middle segment, and the right segment. The repetitions of $e1$ and $e2$ in these segments help the semantic meaning of each segment to be more clear. Intuitively, the middle segment is often more important to reflect the semantic relation. Qin et al. (2016) only used the middle segment with a CNN for RC, while Vu et al. (2016) proposed an extended middle context to pay special attention to the middle part.

Although the middle segment is more significant than two remaining segments in many cases,

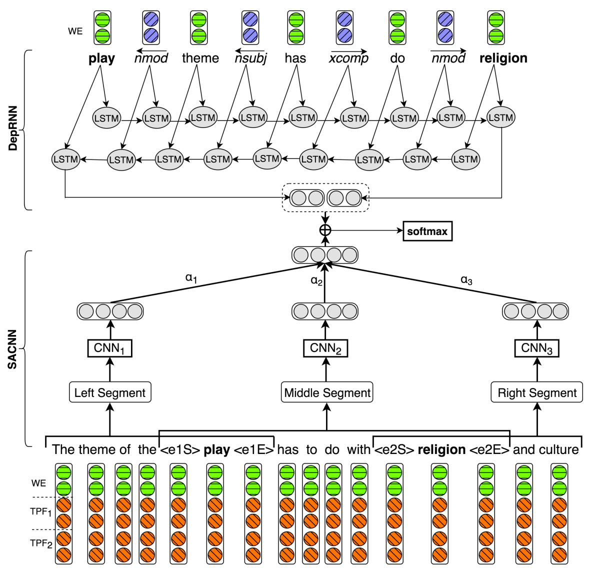

Figure 1: Our model for relation classification.

it is not always true for all. For example, in the sentence "*All other* $\langle e1S \rangle$ ***blood*** $\langle e1E \rangle$ $\langle e2S \rangle$ ***products*** $\langle e2E \rangle$ *are derived from whole blood.*" with the relation label "*Entity-Origin(e2;e1)*", the right segment is more important to reflect the relation type. Besides, the left and right segments might also provide the necessary information to RC. We therefore proceed three segments independently through three separate CNNs, which allow the model to automatically identify segments containing important information. Each CNN includes one convolutional layer and one max-pooling layer.

Let M be a matrix consisting of output vectors of three CNNs: $M = [m_1, m_2, m_3]$, where m_i is the output of CNN_i. The final representation r_1 of the raw sentence generated by SACNN is formed by a weighted sum of output vectors in M:

$$z_i = tanh(m_i),$$
$$\alpha_i = \frac{exp(w^T z_i + b)}{\sum_{i=1}^{3} exp(w^T z_i + b)},$$
$$r_1 = \sum_{i=1}^{3} \alpha_i m_i,$$

where w is a weight vector, w^T is its transformation, and b is a bias parameter.

Dependency-based RNN. While SACNN can learn local features, it cannot handle long-distance dependency between two entities. This disadvantage causes difficulty in correctly assigning subject and object roles of two entities when capturing the directional relation. Meanwhile, RNN could tackle the problem of long-distance pattern learning (Zhang and Wang, 2015). Besides, the SDP naturally offers the relative positions of subjects and objects through the path directions (Xu et al., 2015). We, therefore, exploit SDP based on RNN to gain the information in the directional relation.

An shown in Figure 1, we use Bidirectional Long Short-Term Memory (BLSTM) on the SDP between two entities. Due to its ability to capture long term memory, the BLSTM accumulates increasingly richer information as it goes through the SDP from both two forward and backward directions (Palangi et al., 2016). When it reaches the last two words, the last two hidden states are expected to provide the full semantic meaning of the whole SDP. Additionally, since the length of the SDP is often not so long, we concatenate two output vectors of the last two hidden states as the final representation r_2 of the SDP by DepRNN.

Combination of SACNN and DepRNN. Finally, we combine both SACNN and DepRNN models to exploit fully their own distinct advantages. While SACNN can focus on important segments and gain local features, DepRNN helps to handle long-distance dependency between two entities based on the SDP as well as provide subject and object roles of two entities for the directional relation. Therefore, the final representation r of the relation instance is concatenated by two output vectors (r_1 , r_2) of SACNN and DepRNN, which is then fed to a softmax classifier.

4 Experiments

4.1 Datasets and Settings

We evaluate our model on the SemEval2010 Task 8 which contains $8,000$ training sentences and $2,717$ test sentences, with 19 relations (9 directed relations and an undirected *Other* class). Therefore, the relation classification task is treated as a multi-class classification problem. Following previous work, the official macro-averaged F1-score, which excludes the *Other* relation, is used for evaluation.

We randomly held out 10% of the training set for validation. The Stanford Parser is also used to convert sentences to dependency trees.

For word embeddings, we use the 300-dimensional embeddings of Komninos and Manandhar (2016). In this work, we do not focus on comparing the effectiveness of the different pre-

SACNN	F1
WE, ETF	83.9
WE, TPF	84.5
WE, ETF, TPF	**85.1**

Table 1: Comparison of different features in SACNN. WE, ETF, and TPF stand for Word Embedding, Entity Tag Feature, and Tree-based Position Feature.

Model	Input	F1
CNN	Original Sentence	83.5
CNN	Middle Segment	84.1
SACNN	Three Segments	**85.1**

Table 2: Effectiveness of the segment-level attention.

trained embedding sets. The above pre-trained embedding set is selected since it embeds dependency context to provide valuable syntactic information. Four tokens: $\langle e1S \rangle$, $\langle e1E \rangle$, $\langle e2S \rangle$, $\langle e2E \rangle$ and out-of-vocabulary words are initialized by sampling from a uniform distribution (Kim, 2014). TPF is 15-dimensional and initialized randomly. Thus, the representation of each word has a dimensionality of 330 in the raw sentence.

Hyper-parameters in our model are as follows: 100 filters for each window size [3, 4, 5] and ReLU as the activation function for each CNN in SACNN. In DepRNN, the dimension of each token is 300, the $tanh$ activation function is applied to the last two hidden states, the dimension of each hidden state vector is 150. Other parameters include: L2 regularization with a weight of 10^{-4}, a mini-batch size of 64, a dropout rate at the final layer p = 0.5 before a softmax classifier.

4.2 Results and Analysis

Impact of SACNN and DepRNN. We consider the performance of each model by feeding separately their output vector to a softmax classifier.

In Table 1, we see the effect of different features to SACNN's performance. Combining ETF and TPF significantly enhances the $F1$ score by 0.6%. It proves that ETF and TPF complement each other to more fully provide information about the marked entities and important words to SACNN.

We also examine the segment-level attention mechanism of SACNN. In Table 2, with the same input features (WE, ETF, TPF), the segment-level attention mechanism makes a great contribution by increasing the $F1$ score by 1%.

To check the effect of combining SACNN and DepRNN, in Table 3, we compare the performance of each model to our combined model. First, the SACNN's performance is superior to the DepRNN. One possible reason is that while SACNN selectively focuses on the important segments as well as gains local features from the raw sentences, DepRNN based on the SDP, which is short in the SemEval2010 Task 8, can only provide effectively the entities role. Then, by combining SACNN and DepRNN, our model can exploit the fully necessary information and achieve the best performance.

Model	F1
DepRNN	83.8
SACNN	85.1
Combined	**85.8**

Table 3: Evaluation of our combined model.

Model	Features	F1
SVM (Rink and Harabagiu, 2010)	Rich features	82.2
BLSTM+Attention (Zhou et al., 2016)	WE, ETF	84.0
PECNN (Yang et al., 2016)	WE, DT, TPF, *POS, NER, WordNet*	84.6
CNN+BLSTM (Wang et al., 2017)	WE, DT, PF, *POS, GR, NER, WordNet*	84.7
SR-BRCNN (Wen, 2017)	WE, DT, *POS, NER, WordNet*	85.1
CNN+BLSTM (Zhang et al., 2018)	WE, PF	83.7
CNN+BLSTM+Attention Our model	WE, DT, TPF, ETF	**85.8**

Table 4: Comparison of different classification models. DT, PF stand for Dependency Tree, Position Feature. The *italic* features are external lexical features used.

Comparisons with the State of the Art. We compare our model to some recent work on RC in Table 4. Most previous work exploited some external lexical features (WordNet, NER) and combine NNs to improve the performance (Yang et al., 2016; Wang et al., 2017). Wang et al. (2017) and Wen (2017) proposed complex structures for integrating the CNN and the LSTM, and achieved an $F1$ of 84.7% and 85.1% respectively. Zhang et al. (2018) combined CNN and BLSTM, and reached an $F1$ of 83.7% using only WE, PF features.

Without using any external lexical resources, our model achieves an $F1$ of 85.8%, showing that combining SACNN and DepRNN is very effective, since SACNN helps to selectively focus on the important segments and gains local features, DepRNN provides the role information of subject and object of two entities in addressing the relation directionality. Comparing with some recent work, our model obtains a notable performance.

5 Conclusion

This work presents a new model that combines the SACNN and the DepRNN for RC. Combining ETF and TPF provides entity and semantic information of the input sentences to the model effectively. We also propose the SACNN which automatically focus on the essential segments and gains local features. Besides, the DepRNN helps to exploit long-distance dependency between two entities and their roles. Finally, combining the SACNN and the DepRNN brings the best performance since they highly complement each other. Our model achieved a notable performance on the SemEval2010 Task 8 without using any external lexical resources.

Acknowledgments

This work was partly supported by JST CREST Grant Number JPMJCR1513, Japan. We are grateful to the members of the Computational Linguistics Laboratory, NAIST and the anonymous reviewers for their insightful comments.

References

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Nanda Kambhatla. 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.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Alexandros Komninos and Suresh Manandhar. 2016. Dependency based embeddings for sentence classification tasks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1490–1500.

Hoang-Quynh Le, Duy-Cat Can, Sinh T. Vu, Thanh Hai Dang, Mohammad Taher Pilehvar, and Nigel Collier. 2018. Large-scale exploration of neural relation classification architectures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2266–2277, Brussels, Belgium. Association for Computational Linguistics.

Yann LeCun, Yoshua Bengio, et al. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995.

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

Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 39–48.

Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(4):694–707.

Pengda Qin, Weiran Xu, and Jun Guo. 2016. An empirical convolutional neural network approach for semantic relation classification. *Neurocomputing*, 190:1–9.

Bryan Rink and Sanda Harabagiu. 2010. Utd: Classifying semantic relations by combining lexical and semantic resources. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 256–259. Association for Computational Linguistics.

Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schütze. 2016. Combining recurrent and convolutional neural networks for relation classification. *arXiv preprint arXiv:1605.07333*.

Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. 2016. Relation classification via multi-level attention cnns.

Mengqiu Wang. 2008. A re-examination of dependency path kernels for relation extraction. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.

Pengfei Wang, Zhipeng Xie, and Junfeng Hu. 2017. Relation classification via cnn, segmented max-pooling, and sdp-blstm. In *International Conference on Neural Information Processing*, pages 144–154. Springer.

Ji Wen. 2017. Structure regularized bidirectional recurrent convolutional neural network for relation classification. *arXiv preprint arXiv:1711.02509*.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015. Semantic relation classification via convolutional neural networks with simple negative sampling. *arXiv preprint arXiv:1506.07650*.

Yunlun Yang, Yunhai Tong, Shulei Ma, and Zhi-Hong Deng. 2016. A position encoding convolutional neural network based on dependency tree for relation classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 65–74.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344.

Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*.

Min Zhang, Jie Zhang, Jian Su, and Guodong Zhou. 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.

Xiaobin Zhang, Fucai Chen, and Ruiyang Huang. 2018. A combination of rnn and cnn for attention-based relation classification. *Procedia computer science*, 131:911–917.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 207–212.