Neural Relation Classification with Text Descriptions

Feiliang Ren, Di Zhou, Zhihui Liu, Yongcheng Li, Rongsheng Zhao, Yongkang Liu, Xiaobo Liang
School of Computer Science and Engineering, Northeastern University, Shenyang, 110819, China
renfeiliang@cse.neu.edu.cn

Abstract

Relation classification is an important task in natural language processing fields. State-of-the-art methods usually concentrate on building deep neural networks based classification models on the training data in which the relations of the labeled entity pairs are given. However, these methods usually suffer from the data sparsity issue greatly. On the other hand, we notice that it is very easy to obtain some concise text descriptions for almost all of the entities in a relation classification task. The text descriptions can provide helpful supplementary information for relation classification. But they are ignored by most of existing methods. In this paper, we propose DesRC, a new neural relation classification method which integrates entities text descriptions into deep neural networks models. We design a two-level attention mechanism to select the most useful information from the *"intra-sentence"* aspect and the *"cross-sentence"* aspect. Besides, the adversarial training method is also used to further improve the classification performance. Finally, we evaluate the proposed method on the SemEval 2010 dataset. Extensive experiments show that our method achieves much better experimental results than other state-of-the-art relation classification methods.

1 Introduction

The aim of relation classification is that given a sentence in which two target entities are labeled, to select a proper relation for these two entities from a predefined relation set. For example, given a sentence "*The system as described above has its greatest application in an arrayed <e1>configuration </e1> of antenna <e2>elements</e2>*", a relation classification system aims to identify that there is a "*Component-Whole*" relation from *e2* to *e1*. Obviously, accurate relation classification results would benefit lots of natural language processing tasks, such as sentence interpretations, Q&A, knowledge graph construction, ontology learning, and so on. Thus, lots of researchers have devoted to this research field.

For relation classification, deep neural networks (DNN) based methods have been widely explored and have achieved state-of-the-art experimental results. However, when evaluating state-of-the-art relation classification methods on some standard datasets, experimental results show that there are usually huge performance gaps between different relations. Besides, when using the trained relation classification models to predict relations on new data, the prediction performance is usually far lower than expected. This is mainly because of the data sparsity issue: first, there are always some relations that have far less training data than others; second, the available training data is not sufficient enough to train a robust relation classification model.

On the other hand, we notice that for almost all of the entities in a relation classification task, there are usually available text descriptions for them on some Encyclopedia websites like Wikipedia, DBpedia, Wikidata, etc. For example, from the Wikipedia website, we can extract the following text descriptions as shown in Figure 1, where the example sentence is taken from the SemEval 2010 dataset.

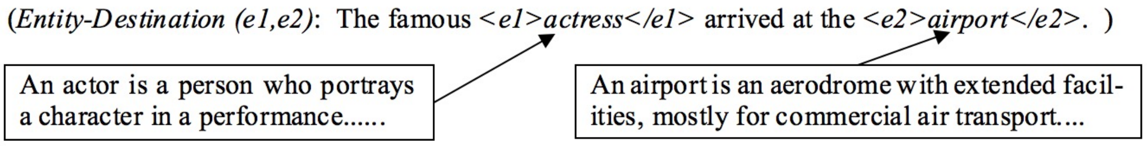
(Entity-Destination (e1,e2): The famous <e1>actress</e1> arrived at the <e2>airport</e2>.)
An actor is a person who portrays a character in a performance......
An airport is an aerodrome with extended facilities, mostly for commercial air transport....

Figure 1: Example of text descriptions extracted from Wikipedia.

2 Related Work

Up to now, lots of novel relation classification methods have been proposed. Early research mainly focuses on features based methods. Usually, these methods firstly select some syntactic and semantic features from the given sentences. Then the selected features are fed into some classification models like support vector machines, maximum entropy, etc.

Recently, DNN based methods have been widely explored and have achieved state-of-the-art experimental results. The core of these methods is to embed features into real-valued vectors, and then feed these vectors into some DNN based learning frameworks. Generally, there are three widely used DNN frameworks for relation classification: convolutional neural networks (CNN), recurrent neural networks (RNN), and their combination. In most recent years, inspired by both the success of DNN methods and the broad consensus that syntactic tree structures are of great help for relation classification, more and more research attention is being paid to the methods that integrate syntactic tree features into DNN based learning frameworks. Among the syntactic tree features, the *Shortest Dependent Path* (SDP) is one of the most frequently used. In Table 1, we summarize some representative state-of-the-art DNN based relation classification methods.

Learning Frameworks		Representative Methods	External resources	Loss function	Optimization method
CNN		Zeng et al., 2014	WordNet	Cross entropy	SGD
		Dos Santos et al., 2015	No	Ranking loss	
DNN+SDP	RNN+SDP	Xu et al., 2016	WordNet	Cross entropy	
	LSTM+SDP	Xu et al., 2015a			
	CNN+SDP	Xu et al., 2015b			
Combination	SDP+ CNN+RNN	Cai et al., 2016			AdaDelta
		Liu et al., 2015			NoReported
	CNN+RNN	Vu et al., 2016	No	Ranking loss	SGD
DNN+Attention	CNN+ Attention	Wang et al., 2016		Distance based loss	
	LSTM+ Attention	Zhou et al., 2016		Cross entropy	AdaDelta
End-to-End Joint Learning		Miwa et al., 2016	WordNet		Adam

Table1: A summarization of representative state-of-the-art relation classification methods.

From Table 1 we can see that there are many similarities among the state-of-the-art relation classification methods. For example, most of them use a cross entropy loss function, use WordNet, and use the stochastic gradient descent (SGD) method for optimization, etc. The main differences among them mainly lie in the learning frameworks.

CNN is a very popular learning framework for relation classification and lots of methods are based on it. For example, Zeng et al. (2014) proposed a CNN based approach for relation classification. In their method, sentence level features are learned through a CNN model that takes word embedding features and position embedding features as input. In parallel, lexical level features are extracted from some context windows that are around the labeled entities. Then the sentence level features and the lexical level features are concatenated into a single vector. This vector is then fed into a *softmax* classifier for relation prediction. Another representative CNN based relation classification method is CR-CNN

(Dos Santos et al., 2015), which tackles the relation classification task with a CNN model that performs classification by ranking. They proposed a new pairwise ranking loss function that is easy to reduce the impact of the artificial relation "*Other*". Their method is also the unique one that takes a specific processing strategy for the "*Other*" relation.

Xu et al. (2016) pointed out that compared with a raw word sequence or a whole parse tree, the SDP between two entities has two main advantages. First, it reduces irrelevant information; second, grammatical relations between words focus on the action and agents in a sentence and are naturally suitable for a relation classification task. Thus many researchers integrate SDP into DNN based learning frameworks for relation classification. For example, based on SDP, Xu et al. (2016) proposed deep recurrent neural networks (DRNNs) for relation classification. Their method can be roughly regarded as a "*RNN + SDP*" relation classification method. Xu et al. (2015a) proposed a neural relation classification architecture that picks up heterogeneous information along the left and right sub-path of the *SDP* respectively, leveraging RNN with multichannel *long short term memory* (LSTM) units. And their method can be roughly regarded as a "*LSTM + SDP*" relation classification method. Other similar work, Xu et al. (2015b) proposed to learn more robust relation representations from SDP through a CNN model; Liu et al. (2015) proposed augmented dependency path (ADP), which is a variant of SDP. Both of these two methods can be roughly regarded as a "*CNN + SDP*" relation classification method.

Some researchers combine CNN and RNN together for relation classification. For example, Vu et al. (2016) investigated CNN and RNN as well as their combination for relation classification. They proposed extended middle context, a new context representation for the CNN architecture. The extended middle context uses all parts of the sentence (the relation arguments, left/right and between of the relation arguments) and pays special attention to the middle part. Meanwhile, they proposed a connectionist bi-directional RNN model and introduced a ranking loss function for the RNN model. Finally, CNN and RNN are combined with a simple voting scheme. Cai et al. (2016) proposed a bidirectional neural network BRCNN, which consists of two RCNNs that can learn features along SDP inversely at the same time. Specifically, information of words and dependency relations is extracted by a two-channel RNN model with LSTM units. The features of dependency units in a SDP are extracted by a convolution layer. Liu et al. (2015) used a RNN model to learn the features of the sub-trees, and used a CNN model to capture the most important features on a SDP.

Recently, the attention method is achieving more and more research attention. Some researchers also add the attention method in their relation classification models. For example, Wang et al. (2016) proposed a multi-level attention CNN model for relation classification. In their method, two levels of attentions are used in order to better discern patterns in heterogeneous contexts. Zhou et al. (2016) proposed an attention-based bidirectional LSTM model for relation classification.

Another research line explores a kind of end-to-end method for relation classification. For example, Miwa et al. (2016) proposed a novel end-to-end neural model to extract entities and the relations between them. Their model captures both word sequence and dependency tree substructure information by stacking bidirectional tree-structured LSTM-RNNs on bidirectional sequential LSTM-RNNs, which allows the model to jointly represent both entities and relations with shared parameters in a single model.

3 Our Model

Figure2 demonstrates the architecture of our method. For each original training/test sentence S_i, it will be augmented to a new triplet format like $<S_i$, $Des_i(e_1)$, $Des_i(e_2)>$, where $Des_i(e_1)$ and $Des_i(e_2)$ are the text descriptions of the labeled entities e_1 and e_2 in S_i. Our model takes the augmented training/test sentences as input.

From Figure2 we can see that in our model, there are three parallel DNN-based encoders that are used to learn the real-valued vector representations for S_i, $Des_i(e_1)$, and $Des_i(e_2)$ respectively. Then with a "*cross-sentence*" attention method, the three learned vector representations are combined into one global real-valued vector representation. Finally, the classification decision is made based on the global vector representation.

For the description representation learning, we use a CNN-based method. And for the original training/test sentence representation learning, we use two methods: one is the CNN-based method that is the

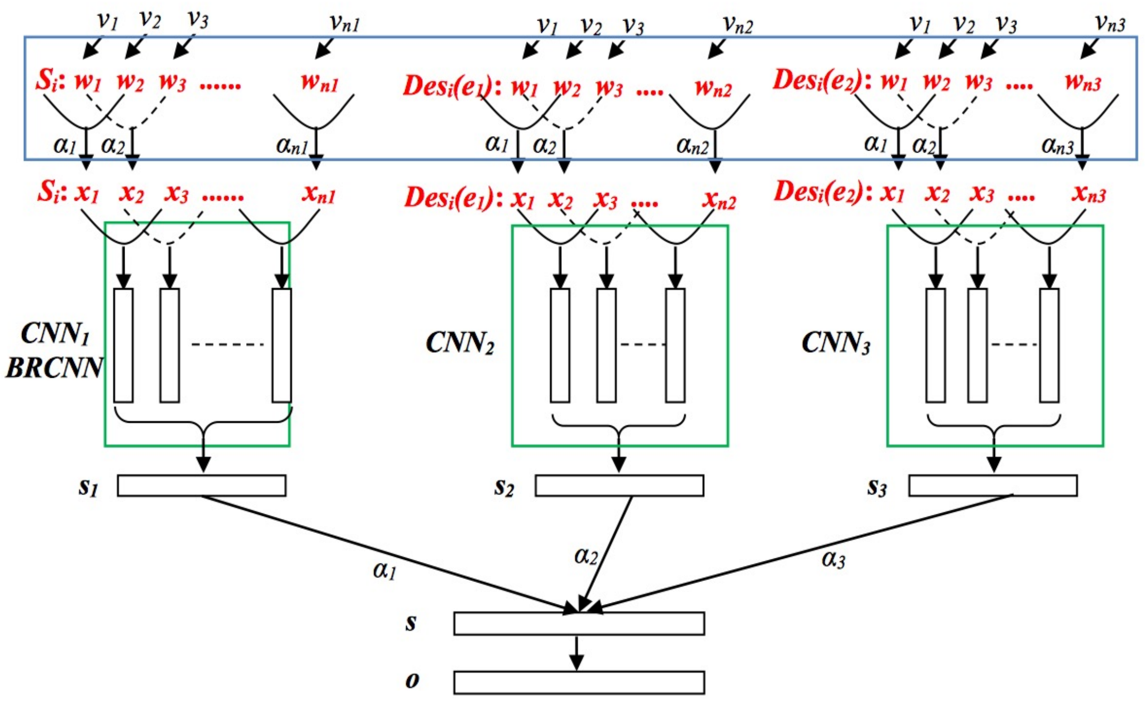
S_i: w_1 w_2 w_3 ...... w_{n1}
$Des_i(e_1)$: w_1 w_2 w_3 .... w_{n2}
$Des_i(e_2)$: w_1 w_2 w_3 .... w_{n3}
S_i: x_1 x_2 x_3 ...... x_{n1}
$Des_i(e_1)$: x_1 x_2 x_3 .... x_{n2}
$Des_i(e_2)$: x_1 x_2 x_3 .... x_{n3}
CNN_1
BRCNN
CNN_2
CNN_3
s_1
s_2
s_3
s
o

Figure 2: Architecture of DesRC

same as the one used in the description representation learning, and the other is a BRCNN-based method that is similar to the one proposed by Cai et al. (2016). In the three representation learning encoders, an "*intra-sentence*" attention method is used to select the most useful word information inside an input sentence. Besides, the adversarial training technique is also used on the word embedding level during training (in Figure2, v_i denotes the adversarial perturbation w.r.t. a word embedding w_i).

3.1 CNN-Based Encoder

Given a sentence, this encoder aims to transform it into a distributed representation via a CNN model. There is a same encoding process for both S_i and its two text descriptions $Des_i(e_1)$ and $Des_i(e_2)$. Here we take S_i as an example to demonstrate the whole CNN-based endoding process. First, each word in S_i is transformed into a real-valued vector representation. Then, a convolutional operation, a max-pooling operation, and a non-linear transformation operation are performed in turn. Finally, the encoder outputs a distributed representation for S_i.

Word Representation Given a sentence S_i = (w_1, w_2, ..., w_n), we transform each of its word w_i into the concatenation of two kinds of embedding representations: 1) a word embedding that captures syntactic and semantic meaning of this word; and 2) a position embedding that specifies which input words are the labeled entities (or entity) or how close an input word to the labeled entities (or entity). Finally, the sentence S_i can be represented as a vector sequence $\boldsymbol{S}_i$ = ($\boldsymbol{w}_1$, $\boldsymbol{w}_2$, ..., $\boldsymbol{w}_n$), where $\boldsymbol{w}_i \in R^d$ and $d = d_a$ + 2*d_b (for original training/test sentences) or $d = d_a + d_b$ (for descriptions). d_a and d_b are the dimension of word embeddings and position embeddings respectively.

***Intra-sentence* Attention** In a sentence, not all of its words are equally useful for the final classification decision. Some of them may be important, some of them may not. Thus, we design an "*intra-sentence*" attention method to automatically identify which words in a sentence are more important for the final classification decision. Following Feng et al. (2017), for each word $\boldsymbol{w}_i$ in $\boldsymbol{S}_i$, we take the embedding of its m context words and the corresponding entities' embeddings as input, the "*intra-sentence*" attention model outputs a new representation $\boldsymbol{x}_i$ for $\boldsymbol{w}_i$ with the following formula.

$$x_i = \sum_{l=1}^{w} \alpha_l * \boldsymbol{m}_l \qquad (1)$$

where $\boldsymbol{m}_l \in R^d$ is the embedding of a considered context word, and the attention score α_l is defined as:

$$\alpha_l = \frac{exp(f_i)}{\sum_{j=1}^{n} exp(f_j)} \tag{2}$$

f_i is a function to evaluate the semantic relatedness of a text fragment with the given entity (for text descriptions) or the given entity pair (for the original training/test sentences) and the linked relation. It is calculated with the following formulas.

$$f_i = tanh(\boldsymbol{M} * [\boldsymbol{m_i}; \boldsymbol{w_{e1}}; \boldsymbol{w_{e2}}] + U * \boldsymbol{r}) \tag{3}$$

$$or \quad f_i = tanh(\boldsymbol{M_{1/2}} * [\boldsymbol{m_i}; \boldsymbol{w_{e1}}/\boldsymbol{w_{e2}}] + U * \boldsymbol{r}) \tag{3'}$$

where $\boldsymbol{M} \in R^{d*3}$, $\boldsymbol{M}_{1/2} \in R^{d*2}$, $\boldsymbol{U} \in R^{dr}$, d_r is the dimension of relation, and $\boldsymbol{w}_{e1/e2}$ and $\boldsymbol{r}$ are the word embeddings of the labeled entities and the relation linked with them. The "[]" symbol denotes the concatenation operation. Similar to Lin et al. (2017), in the testing phase, since $\boldsymbol{r}$ is not known, we will take each possible relation into consideration. Finally, the "*intra-sentence*" attention model outputs a new representation for $\boldsymbol{S}_i$ that can be denoted as $\boldsymbol{S}_i = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)$.

Convolution Transformation After the above operation, a convolutional layer is used to extract the local features of an input sentence. This layer slides a word window of length *w* over the input sentence and performs a convolution operation within each sliding window. Its output for the *i-th* window is computed with the following formula.

$$\boldsymbol{C_i} = \boldsymbol{M_3} * \boldsymbol{cx_i} + \boldsymbol{b_1} \tag{4}$$

where $\boldsymbol{M}_3 \in R^{h1*(d*w)}$, h_1 (a hyper-parameter) is the size of hidden units in the convolutional layer, $\boldsymbol{cx}_i$ is the embedding concatenation of the *w* word within the *i-th* window, and $\boldsymbol{b}_1$ is a bias term.

Max-pooling After the convolutional transformation, a max-pooling operation is applied to capture the most useful local features produced by the convolutional operation. This process can be written as the following formula.

$$\boldsymbol{p_i} = max_n \boldsymbol{C(i, n)}; \qquad 0 \leq i \leq h_1 \tag{5}$$

The result of max-pooling is an h_1-element real-valued vector whose size is no longer related to the length of the input sentence.

Non-linear Operation After the max-pooling operation, its output vectors $\boldsymbol{p}$ is fed into a non-linear transformation layer to generate the final representation for S_i with the following formula.

$$\boldsymbol{s_i} = tanh(\boldsymbol{M_4} * \boldsymbol{p} + \boldsymbol{b_2}) \tag{6}$$

where $\boldsymbol{M}_4 \in R^{h2*h1}$, h_2 (a hyper-parameter) is the size of hidden units in this layer, and $\boldsymbol{b}_2$ is a bias term.

3.2 BRCNN-Based Encoder

BRCNN is a relation classification model that is first proposed by Cai et al. (2016). It builds a relation classification model based on SDP, and can take full advantages of both the CNN model and the RNN model. Specifically, given a sentence and its dependency tree, the BRCNN model first extracts a SDP from the dependency tree. Then, along the SDP, two RNN models with LSTM units are used to learn hidden representations of words and dependency relations respectively. A convolution layer is applied to capture local features from hidden representations of every two neighbor words and the dependency relations between them. A max-pooling layer thereafter gathers information from local features of the SDP and the inverse SDP. Finally, a linear operation is performed to generate the final representation of the given sentence. More detail information can be referred to the work of Cai et al. (2016).

Here we take the BRCNN model as a new encoder to learn the representations of the original training/test sentences mainly for the following two reasons. First, BRCNN is one of the current-best relation classification models, and we want to explore the maximal potential of our method. Second, we want to explore the effectiveness of text descriptions in different relation classification frameworks.

However, it should be noted that the BRCNN-based encoder cannot be used in the description representation learning. This is because that the BRCNN model is based on SDP, which is extracted from the shorted dependency path between the **TWO** labeled entities, but a text description only focuses on **ONE** single entity. Thus it is impossible to extract a SDP from an entity's text description. Accordingly, the BRCNN based encoder cannot be utilized.

3.3 Representation Combination

After the CNN/BRCNN based representation learning, there will be three generated representations that are for the original training/test sentence and its two text descriptions respectively. We use a "*cross-sentence*" attention method to combine them into **ONE** global real-valued representation. With this attention method, we can capture the most useful information among different representations. The global representation ***s*** is computed as a weighted sum of the single representation vectors.

$$\boldsymbol{s} = \sum_i \alpha_i * \boldsymbol{s}_i \tag{7}$$

The attention score α_i is defined as:

$$\alpha_i = \frac{exp(g_i)}{\sum_{j=1}^{k} exp(g_j)} \tag{8}$$

g_i is a function to evaluate how well a representation vector reflects its corresponding entities (or entity) and the relation linked by the two labeled entities. It is calculated with the following formula.

$$g_i = \boldsymbol{s_i} * \boldsymbol{r} \tag{9}$$

where ***r*** is the embedding of the corresponding relation. In the testing phase, we will also take each possible relation into consideration

3.4 Classification Prediction

After the representation combination, the generated representation ***s*** is fed into a linear output layer to compute the confidence score for each possible relation. A *softmax* classifier is further used to get the probability distribution ***y*** over all relations. This process is written as formula10.

$$\boldsymbol{y} = softmax(\boldsymbol{M_5} * \boldsymbol{s} + \boldsymbol{b_3}) \tag{10}$$

where $\boldsymbol{M}_5 \in R^{h3*h2}$ and h_3 is the number of all possible relations.

3.5 Dropout Operation

Over-fitting is an issue that cannot be ignored in DNN models. Hinton et al. (2012) proposed the dropout method that has been proved to be effective for alleviating this issue. This method randomly sets a proportion (called drop rate, a hyper-parameter) of features to zero during training. It is expected to obtain less interdependent network units, thus the over-fitting issue is expected to be alleviated. In our method, we take dropout operations on $\boldsymbol{w}_i$ (see the word representation section) and ***s*** (see formula 7). The drop rates for them are denoted as $dp_{1\sim2}$ respectively.

3.6 Training Procedure

All the parameters in our method can be denoted as θ=($\boldsymbol{E}^w$,$\boldsymbol{E}^p$,$\boldsymbol{M}$,$\boldsymbol{M}_1$,$\boldsymbol{M}_2$,$\boldsymbol{M}_3$,$\boldsymbol{M}_4$,$\boldsymbol{M}_5$,$\boldsymbol{U}$,$\boldsymbol{b}_1$,$\boldsymbol{b}_2$,$\boldsymbol{b}_3$,$\boldsymbol{r}$),where $\boldsymbol{E}^w$ and $\boldsymbol{E}^p$ represent the embedding matrices of word and position respectively. In this paper, we use the *word2vecc* toolkit (Mikolov et al., 2013) to train the word embedding matrix $\boldsymbol{E}^w$ on the English Wikipedia data from May 2014, which is similar to Vu et al. (2016). $\boldsymbol{E}^p$, other transformation matrices, and the bias terms are randomly initialized. All the parameters are tuned using the back propagation method. SGD optimization technique is used for training. Formally, we try to maximize the following loss function.

$$L(\theta) = \sum_{i=1}^{N} log(y_i) \tag{11}$$

where *N* is the total number of training samples. During training, each input sample is considered independently. And each parameter is updated by applying the following update rule, where η is the learning rate.

$$\theta = \theta + \eta * \partial log y_i / \partial \theta \tag{12}$$

3.7 Adversarial Training

Adversarial training (Goodfellow et al., 2014) is a method that adds some random perturbations to the training data, whose aim is to improve the robustness of a classifier. Researchers (Wu et al., 2017, Miyato et al., 2016, etc) show that when some adversarial noise is added at the level of word embedding by computing the gradient direction of a loss function w.r.t. the data, better experimental results are obtained. Following these previous methods, we also add adversarial noise at the word embedding level during training. This process is written as the following formulas.

$$\boldsymbol{w_i} = \boldsymbol{w_i} + \boldsymbol{v_i} \tag{13}$$

$$\boldsymbol{v_i} = \epsilon * g/||g|| \qquad where \quad g = \partial L/\partial w_i \tag{14}$$

where L is the loss function, and $||g||$ is the norm of gradients over all the words in a given training sentence (or the text descriptions).

4 Experimental Results and Analysis

Dataset The SemEval-2010 Task 8 dataset is used to evaluate our method. In this dataset, there are 8000 training sentences and 2717 test sentences. For each training/test sentence, two entities that are expected to be predicted a relation are labeled. In this dataset, there are 9 relations whose directions need to be considered and an extra artificial relation "*Other*" that does not need to consider the direction. Thus totally there are 19 relations in this dataset. Some statistics of this dataset are reported in Table 2. In this paper, macro-averaged *F1* score (excluding "*Other*"), the official evaluation metric, is used. And the direction of a relation is considered. In experiments, we download the needed text descriptions for all the labeled entities from *Wikipedia*[1]. All the dependency parsing trees used in the BRCNN model are generated by the Stanford Parser (Klein and Manning, 2003).

Relations	# in training dataset	# in test dataset
Cause-Effect(e1,e2)	151/5.56%	344/4.3%
Cause-Effect(e2,e1)	207/7.62%	659/8.24%
Component-Whole(e1,e2)	162/5.92%	470/5.88%
Component-Whole(e2,e1)	153/5.63%	471/5.89%
Content-Container(e1,e2)	179/6.59%	374/4.68%
Content-Container(e2,e1)	37/1.36%	166/2.08%
Entity-Destination(e1,e2)	316/11.63%	844/10.55%
Entity-Destination(e2,e1)	0/0%	1/0.01%
Entity-Origin(e1,e2)	223/8.21%	568/7.1%
Entity-Origin(e2,e1)	43/1.58%	148/1.85%
Instrument-Agency(e1,e2)	25/0.92%	97/1.21%
Instrument-Agency(e2,e1)	139/5.12%	407/5.09%
Member-Collection(e1,e2)	42/1.54%	78/0.98%
Member-Collection(e2,e1)	235/8.65%	612/7.65%
Message-Topic(e1,e2)	245/9.02%	490/6.13%
Message-Topic(e2,e1)	62/2.28%	144/1.8%
Product-Producer(e1,e2)	120/4.42%	323/4.04%
Product-Producer(e2,e1)	131/4.82%	394/4.93%
Other	247/9.09%	1410/17.63%

Table 2: Statistics for the Experimental Dataset

In experiments, we apply a cross-validation procedure on the training data to select suitable hyper-parameters. Finally, the best configurations are: the dimensions of word embeddings (d_a) and relation embeddings (d_r) are both set to 300, the dimension of position embeddings (d_b) is set to 15, learning rate η is set to 0.001, $h_{1\sim2}$ are set to 200 and 300, $dp_{1\sim2}$ are set to 0.35 and 0.3, both m (see formula 1) and w (see formula 4) are set to 3, the adversarial training parameter ϵ (see formula 14) is set to 0.01.

[1]https://www.wikipedia.org/

Effectiveness of Different Model Components In the first part of our experiments, we conduct experiments to evaluate: (1) the effectiveness of text descriptions; (2) the contributions of two proposed attention methods; and (3) the contribution of the adversarial training. To this end, we implement two classification models: one is a "*CNN-CNN-CNN*" model that uses the CNN based method for the original training/test sentence representation learning; and the other is a "*BRCNN-CNN-CNN*" model that uses the BRCNN based method for the original training/test sentence representation learning. They are denoted as *DesRC*(CNN) and *DesRC*(BRCNN) respectively. In experiments, we first implement a basic CNN model and a basic BRCNN model. Based on them, we incrementally add text descriptions, attention methods and the adversarial training. The experimental results are reported in Table 3.

Model	F1	Model	F1
CNN	83.6	BRCNN	84.5
+ *adv*	83.9	+ *adv*	84.7
+ "*intra*" *att*	84.7	+ "*intra*" *att*	84.7
+ *des*(e_1) + "*cross*" *att*	84.5	+ *des*(e_1) + "*cross*" *att*	85.1
+ *des*(e_2) + "*cross*" *att*	84.6	+ *des*(e_2) + "*cross*" *att*	84.8
+ *des*(e_1+ e_2) + "*cross*" *att*	85.3	+ *des*(e_1+ e_2) + "*cross*" *att*	85.7
+ *des*(e_1) + "*intra+cross*" *att*	84.7	+ *des*(e_1) + "*intra+cross*" *att*	85.0
+ *des*(e_2) + "*intra+cross*" *att*	84.6	+ *des*(e_2) + "*intra+cross*" *att*	84.9
+ *des*(e_1+ e_2) + "*intra+cross*" *att*	**85.6**	**+ *des*(e_1+e_2) + "*intra+cross*" *att***	**86.3**
+ "*intra*" *att* + *adv*	84.8	+ "*intra*" *att* + *adv*	85.2
+ *des*(e_1) + "*cross*" *att+ adv*	84.7	+ *des*(e_1) + "*cross*" *att+ adv*	85.0
+ *des*(e_2) + "*cross*" *att+ adv*	84.7	+ *des*(e_2) + "*cross*" *att+ adv*	86.1
+ *des*(e_1+ e_2) + "*cross*" *att+ adv*	85.4	+ *des*(e_1+ e_2) + "*cross*" *att+ adv*	86.4
+ *des*(e_1) + "*intra+cross*" *att+ adv*	84.9	+ *des*(e_1) + "*intra+cross*" *att+ adv*	85.4
+ *des*(e_2) + "*intra+cross*" *att+ adv*	84.7	+ *des*(e_2) + "*intra+cross*" *att+ adv*	85.2
+ *des*(e_1+ e_2) + "*intra+cross*" *att+ adv*	**86.1**	**+ *des*(e_1+ e_2) + "*intra+cross*" *att+ adv***	**86.7**
+ *des*(e_1+ e_2)+"*intra+cross*"*att+adv+WN*	**86.6**	**+ *des*(e_1+ e_2) + "*intra+cross*"*att+adv+WN***	**87.4**

Table 3: Performance of *DesRC*(CNN)/*DesRC*(BRCNN)
with different features, WN means WordNet.

From Table 3 we can draw the following conclusions.

First, there is substantial performance improvement for both the basic CNN model and the basic BRCNN model when text descriptions are added. With the "*intra-sentence*" and "*cross-sentence*" attention methods, the F1 score for the basic CNN model increases about 2, and the F1 score for the basic BRCNN increases about 1.8. The experimental results indicate that text descriptions could provide much useful decision information for relation classification. Accordingly, the data sparsity issue is alleviated greatly when text descriptions are utilized. It is worth noting that when only part of the descriptions used (for example, only the descriptions of *e1* or *e2* are used), the F1 score still improves for both the basic CNN model and the basic BRCNN model. This is in line with our original hypothesis: as long as the text description could provide some property information for a given entity, the possible relations linked by this entity will be limited into a small candidate set. Thus, the classification decisions are more easy to be made. Accordingly, the F1 score increases.

Second, a well-designed attention method can improve the performance of relation classification greatly. For example, when using only *des(e1)* or *des(e2)* with an "*cross-sentence*" attention method, the maximal F1 improvement for the basic CNN model and the basic BRCNN model are about 1 and 0.6 respectively. We notice that the "*intra-sentence*" attention contributes much more performance improvement for the basic CNN model (F1 increases 1.1) than for the basic BRCNN model (F1 increases only 0.2). This is because that the basic BRCNN model is based on SDP whose average word number is very small (about 4, Cai et al., 2016). Thus as long as a word is in SDP, it must be much important. In other words, the role of SDP and the attention method overlaps to a certain extent. As a result, the role of the "*intra-sentence*" attention for a basic BRCNN model is not as obvious as for a basic CNN model.

Third, the adversarial training method can further improve the performance of relation classification. When adding the adversarial training method in the basic CNN model and the basic BRCNN model, their F1 scores increase about 0.3 and 0.2 respectively. When both the description information and two attention methods are used, the F1 contributions of the adversarial training method are 0.5 and 0.4 for the basic CNN model and the basic BRCNN model respectively.

Fourth, when WordNet is added, the F1 scores for both *DesRC*(CNN) and *DesRC*(BRCNN) further increase about 0.5 and 0.7, which is a little less than the results reported in other state-of-the-art DNN based methods. In their methods, the F1 score often increases about 1 when WordNet is used. We think this is because that part of the information provided by WordNet can also be provided by text descriptions. Thus, the role of WordNet decreases when text descriptions are used. In other words, in a relation classification task, text descriptions can replace WordNet to a certain extent.

Comparisons with other State-of-the-art Relation Classification Methods In the second part of our experiments, we compare our method with several state-of-the-art DNN based relation classification methods. The comparison results are shown in Table 4. From Table 4 we can see that even without

Methods	Extra resources used	F1
CNN(Zeng et al., 2014)	WordNet	82.7
CRCNN(Dos Santos et al., 2015)	No	84.1
RNN + SDP(Xu et al., 2016)	WordNet	86.1
LSTM + SDP(Xu et al., 2015a)	WordNet	83.7
CNN + SDP(Xu et al., 2015b)	WordNet	85.6
CNN + RNN + SDP(Cai et al., 2016)	WordNet	**86.3**
CNN + SDP(Liu et al., 2015)	WordNet	83.6
CNN + RNN(Vu et al., 2016)	No	84.9
LSTM + Attention(Zhou et al., 2016)	No	84
End-to-End(Miwa et al., 2016)	WordNet	85.5
DesRC(CNN)	Text descriptions	***86.1***
DesRC(BRCNN)	Text descriptions	***86.7***
DesRC(CNN)	Text descriptions + WordNet	***86.6***
DesRC(BRCNN)	Text descriptions + WordNet	***87.4***

Table 4: Comparisons with other state-of-the-art DNN based methods.

WordNet, our method still achieves much better results than the baselines. For *DesRC*(CNN), its F1 score is close to the current-best F1 score very much. And for *DesRC*(BRCNN), it achieves the best F1 score. When WordNet is used, both *DesRC*(CNN) and *DesRC*(BRCNN) outperform the current-best method. These experimental results indicate that text descriptions are of great adaptive. They can benefit both the CNN based learning framework and the RNN based learning frameworks (BRCNN can be seen as a combination of CNN and RNN). What's more, compared with WordNet, text descriptions are more easily obtained. Thus, our method is easier to be transplanted to a new language's relation classification task.

Detailed Results In the third part of our experiments, we compare the classification performance of different relations. Here the used model is *DesRC*(BRCNN) and WordNet is used. The comparison results are reported in Table 5.

From Table 5 we can see that there are still huge performance gaps between different relations in our method. For example, the best F1 score (for example, "*cause-effect*" and "*entity-destination*") is almost 10 higher than the worst F1 score (for example, "*product-producer*" and "*content-container*"). But compared with the huge imbalance among the number of their training samples, the performance gaps are far smaller. These experimental results also show the effectiveness of text descriptions for alleviating the data sparsity issue.

Relations	P	R	F
Cause-Effect	96.43	90.25	93.24
Component-Whole	86.51	82.05	84.22
Content-Container	84.28	80.26	82.22
Entity-Destination	95.40	88.04	91.57
Entity-Origin	89.78	87.64	88.70
Instrument-Agency	87.97	81.61	84.67
Member-Collection	91.71	82.23	86.71
Message-Topic	94.05	83.28	88.34
Product-Producer	85.68	80.88	83.21

Table 5: Classification results of different relations

5 Conclusions and Future Work

In this paper, we propose a new relation classification method that uses text descriptions as a kind of supplement information. The main contributions of our method are listed as follows.

First, to the best of our knowledge, this is the first relation classification method that uses description information. In our method, two well-designed attention methods are used to combine the classification features that come from the original input sentences and their corresponding descriptions. Besides, the adversarial training method is also used to further improve the performance of our method.

Second, we conduct extensive experiments to evaluate the proposed method. Experimental results show that our method is much effective for relation classification, and it outperforms all the compared state-of-the-art baselines.

In the future, we will further explore the following two research directions. First, we will explore more kinds of descriptions that come from different websites, even to explore multi-lingual descriptions, as Lin et al. (2017) do in their work. Second, we will explore whether there are more effective learning framework that can take personalized classification strategies for different relations.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (NSFC No. 61572120, 61672138 and 61432013).

References

Nguyen Bach and Sameer Badaska. 2007. A review of relation extraction. Literature review for Language and Stastics H.

Ronan Collobert, Jason Weston, Lon Bottou, Michael Karlen, KorayKavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, pages 12:2493–2537.

Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceedings of the 53th Annual Meeting of the Association for Computational Linguistics*, pages 626–634.

Xiaocheng Feng, Jiang Guo, Bing Qin, Ting Liu, and Yongjie Liu. 2017. Effective deep memory networks for distant supervised relation extraction. In *Proceedings of IJCAI-17*, pages 4002–4008.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*.

Kazuma Hashimoto, Makoto Miwa, YoshimasaTsuruoka, and Takashi Chikayama. 2013. Simple customization of recursive neural networks for semantic relation classification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1372–1376.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. In *proceedings of 2003 ACL*, page arXiv: 1207.0580.

Dan Klein and Christopher D.Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of 2003 ACL*, pages 432–430.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 2124–2133.

Yang Liu, Furu Wei, Sujian Li, HengJi, Ming Zhou, and Houfeng Wang. 2015. A dependency-based neural network for relation classification. In *Proceedings of the 53th Annual Meeting of the Association for Computational Linguistics*, pages 285–290.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the Workshop at ICLR*.

T. Miyato, A. M. Dai, and I. Goodfellow. 2016. Adversarial Training Methods for Semi-Supervised Text Classification. *ArXiv e-prints*, May.

RuiCai, Xiaodong Zhang, and Houfeng Wang. 2016. Bidirectional recurrent convolutional neural network for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 756–765.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y.Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on EMNLP and Computational Natural Language Learnin*, pages 1201–1211.

Richard Socher, Alex Perelygin, Jean Y.Wu, Christopher D.Manning Jason Chuang, Andrew Y.Ng, and Christopher Potts. 2013a. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

Richard Socher, John Bauer, Christopher D.Manning, and Andrew Y.Ng. 2013b. Parsing with compositional vector grammars. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics*, pages 455–465.

S.Petrov and D.Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of NAACL HLT 2007*, pages 404–411.

S.Petrov and D.Klein. 2017. Neural relation extraction with multi-lingual attention. In *Proceedings of the 55th ACL*, pages 34–43.

Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and HinrichSchutze. 2016. Combining recurrent and convolutional neural networks for relation classification. In *Proceedings of NAACL-HLT 2016*, pages 534–539.

Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. 2016. Relation classification via multi-level attention cnns. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1298–1307.

Yi Wu, David Bamman, and Stuart Russell. 2017. Adversarial training for relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1779–1784.

Kun Xu, Yangsong Feng, Songfang Huang, and Dongyan Zhao. 2015a. Semantic relation classification via convolutional neural networks with simple negative samplin. In *Proceedings of 2015 Conference on Empirical Methods in Natural Language Processing*, pages 536–540.

Yan Xu, LiliMou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015b. Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1785–1794.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2015. Relation classification via convolutional deep neural network. In *Proceedings of the 25th International Conference on Computational Linguistics*, pages 2335–2344.

Zhisong Zhang, Hai Zhao, and Lianhui Qin. 2016. Probabilistic graph-based dependency parsing with convolutional neural network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1382–1392.