Joint Learning of Entity Semantics and Relation Pattern for Relation Extraction

Suncong Zheng, Jiaming Xu(✉), Hongyun Bao, Zhenyu Qi, Jie Zhang, Hongwei Hao, and Bo Xu

Institute of Automation, Chinese Academy of Sciences, Beijing 100190, People's Republic of China
{suncong.zheng,jiaming.xu,hongyun.bao,zhenyu.qi, jie.zhang,hongwei.hao,bo.xu}@ia.ac.cn

Abstract. Relation extraction is identifying the relationship of two given entities in the text. It is an important step in the task of knowledge extraction, which plays a vital role in automatic construction of knowledge base. When extracting entities' relations from sentences, some keywords can reflect the relation pattern, besides, the semantic properties of given entities can also help to distinguish some confusing relations. Based on the above observations, we propose a mixture convolutional neural network for the task of relation extraction, which can simultaneously learn the semantic properties of entities and the keyword information related to the relation. We conduct experiments on the SemEval-2010 Task 8 dataset. The method we propose achieves the state-of-the-art result without using any external information. Additionally, the experimental results also show that our approach can learn the semantic relationship of the given entities effectively.

Keywords: Relation extraction · Convolutional neural network · Entity embedding · Keywords extraction

1 Introduction

Relation extraction is identifying semantic relation of the entity pairs in a sentence, which is also called relation classification. It serves as an intermediate step in knowledge extraction from unstructured texts, which plays an important role in automatic knowledge base construction

Classical methods for the task of relation extraction focus on designing effective handcrafted features to obtain better classification performance [1,2,9]. These handcrafted features are extracted by analyzing the text and using different natural language processing (NLP) tools. However, these methods need complicated feature engineering and heavily rely on the supervised NLP toolkits, which might lead to the error propagation. In order to reduce the manual work in feature extraction, recently, deep neural networks [3,5–7] have been applied to obtain effective relation features from sentences directly. Although these models

P. Frasconi et al. (Eds.): ECML PKDD 2016, Part I, LNAI 9851, pp. 443–458, 2016.
DOI: 10.1007/978-3-319-46128-1_28

Table 1. Instances in the SemEval-2010 Task 8 dataset.

Entity-Destination(e1,e2):
(1) Mayans charted venuss motion across the sky poured $[chocolate]_{e1}$ **into** $[jars]_{e2}$ and interred them with the dead
(2) Both his $[feet]_{e1}$ have been **moving into** the $[ball]_{e2}$ union members
Cause-Effect(e2,e1):
(3) Plantar $[warts]_{e1}$ are **caused by** a $[virus]_{e2}$ that infects layer of skin
(4) A wind speed associated with the $[devastation]_{e1}$ **caused by** the $[tornado]_{e2}$
Other:
(5) Frequent agitations throw academic $[life]_{e1}$ **into** $[disarray]_{e2}$
(6) Painting shows a historical view of the $[damage]_{e1}$ **caused by** the 1693 catania earthquake and the $[reconstruction]_{e2}$

can learn related features from given sentences without complicated feature engineering work, most of them focus on learning the semantic representation of the whole sentence, and they pay a little attention to keyword information related to the relation. Besides, they also fail to take full advantage of the entities' semantic properties.

Based on our observations, we find that most relation pattern can be reflected by some keywords in a sentence, especially the words between the given entities. We randomly select some instances from the SemEval-2010 Task 8 dataset [9] as Table 1 shows. If entity *e1* and entity *e2* satisfy the relation of "*Entity-Destination(e1,e2)*", the words between *e1* and *e2* may be direction words such as: "into". If given entities satisfy the relation of "*Cause-Effect(e2,e1)*", the words between *e1* and *e2* are more inclined to past participles such as: "caused by". Therefore, when compared with learning a semantic embedding of the whole sentence, extracting keyword information between given entities can better reflect the relation pattern in the sentence. Convolutional neural networks (CNN) [11–13] have achieved great success in sentence's semantic representation. It is able to preserve sequence information and extract the keyword information in a sentence. Therefore, in order to extract the keyword information which can reflect the relation pattern, we adopt a CNN architecture to model the sub-sentence between given entities instead of modeling the whole sentence.

However, it is hard to distinguish the confusing relation category by only using the keyword information. As Table 1 shows, the sub-sentence between *e1* and *e2* in sentence *5* seems to describe the relation of "*Entity-Destination(e1,e2)*", but the semantic information of given entities show that they do not have the relationship. Hence, making good use of given entities' semantic properties can help to distinguish the confusing relationship. The semantic properties of given entities can be reflected by their contextual words. Some entities' properties may be reflected by the former (next) one word, some may be reflected by the former (next) two words or more. We set the entity word as the center, and select different sub-sentences around the entity as the entity's

contexts. Extracting the entity's semantic properties is mining the semantic information of these contextual sub-sentences. To achieve this, we apply the operation of mixture convolution to extract the entities' different contextual features, then use max-pooling operation to select the most suitable contextual features as the entity's semantic properties. In this manner, we can solve the problem of unknown entity words, and represent the semantic relationship of entities effectively.

Therefore, the model we propose, in this paper, is a kind of mixture convolutional neural network, which can simultaneously learn the semantic properties of entities and the keyword information related to the relation. Different components of the mixture model focus on extracting different information, and all information is merged in the output layer to fix the task of relation classification.

The main work and contributions of this paper can be summarized as follows: (1) We propose a mixture convolutional neural network for the task of relation classification, which can simultaneously learn the semantic properties of entities and the keyword information related to the relation. (2) The method we proposed achieves the state-of-the-art results on the SemEval-2010 Task 8 dataset without using any external information such as: Word-Net or NLP tools. (3) We also conduct experiments to analyze the entity embedding produced by our method. The experimental results also show that our approach can represent semantic relationship of given entities effectively, when compared with word2vec [14].

2 Related Work

Over the years, relation classification is a widely studied task in the NLP community. To accomplish the task, various approaches have been proposed. Existing methods for relation classification can be divided into handcrafted feature based methods [1,2], neural network based methods [3–7] and the other valuable methods [6,10].

The handcrafted feature based methods focus on using different natural language processing (NLP) tools and knowledge resources to obtain effective handcrafted features. Then, they use some statistical classifier such as Support Vector Machines (SVM) [23] or Maximum Entropy (MaxEnt) [24] to get the right relation class based on the handcrafted features. The early work [2] employs Maximum Entropy model to combine diverse lexical, syntactic and semantic features derived from the text. Rink et al. [1] further designs 16 kinds of features that are extracted by using many supervised NLP toolkits and resources. It can get the best result at SemEval-2010 Task 8 when compared with other handcrafted features based methods.

In recent years, deep neural models have made significant progress in the task of relation classification. These models learn effective relation features from the given sentence without complicated feature engineering. The most common neural-network based models applied in this task are Convolutional Neural Networks (CNN) [3,4,8] and sequential neural networks such as Recursive Neural Networks (RecNN) [7] and Long Short Term Memory Networks (LSTM) [5].

Zeng [3] early explores convolutional neural network to represent the sentence level features. But the method still need to use features derived from lexical resources such as Word-Net to achieve the state-of-the-art results. Santos [4] and Xu [8] also apply convolutional neural network to classify relation classes. Santos [4] uses a pair-wise ranking method instead of softmax function on the top of CNN to reduce the effect of the confusing relation "Other" and Xu [8] proposes a negative sampling strategy to improve the assignment of subjects and objects. Some other deep learning approaches [5,7] focus on learning the whole sentences' semantic representation. Their differences mainly concentrate in the model architectures they used. There also exists other valuable methods such as the kernel-based methods [10] and the compositional embedding model [6].

In this paper, we find that keyword information between given entities and the semantic properties of given entities are important factors to reveal relationship. Therefore, we propose a kind of mixture convolutional model by joint learning the entity semantic properties and relation keywords for the task of relation classification.

3 Our Method

In order to extract the relation pattern and reduce the effect of confusing relations for the task of relation classification, we propose the mixture convolutional neural network (MixCNN), an unified model by joint learning of entities' semantic properties and relation pattern. In the following sections, we firstly present the architecture of our method shown in Fig. 1 and then detail each component of the model. After that, we introduce the objective function and training details.

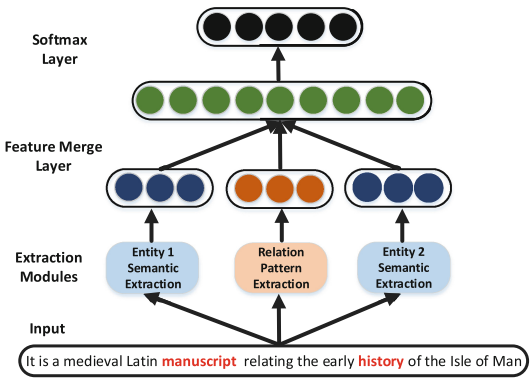

Fig. 1. The architecture of mixture convolutional neural network.

3.1 The Architecture of MixCNN

The framework of MixCNN is shown in Fig. 1, which mainly contains the entity semantic extraction module (ESE) and the relation pattern extraction module (RPE). When given a sentence, the entity semantic extraction module focuses on extracting the semantic properties of the given entities based on their surrounding words. The relation pattern extraction module focuses on extracting the keyword information between the two given entities which can reflect the relation pattern [4]. We merge the information of entities and relation pattern, obtained from the ESE and RPE modules, then fed the merged information into a softmax layer to fix the task of relation classification. In what follows, we describe these modules in detail.

3.2 The Module of Relation Pattern Extraction

Based on our observations and Santos's analysis [4], we find that most relation patterns can be reflected by a few keywords between the given two entities. Hence, the module of RPE aims to extract the keyword information which is related to the target relation. Convolutional neural network (CNN) [11–13] is able to preserve the sequence information and extract the keyword information in a sentence. [3,4,8] also validate the effectiveness of CNN to extract the related keyword information. Therefore, in order to extract the keyword information which can reflect relation pattern, we adopt the CNN architecture [12] to model the sub-sentence between the given entities instead of representing the whole sentence as Fig. 2 shows.

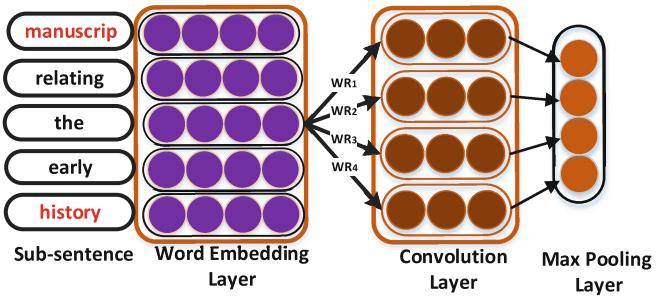

Fig. 2. The module of relation pattern extraction.

Firstly, each word is represented by a word embedding. In our experiments, we employ the word2vec[1] [14] to produce the word embeddings based on Wikipedia corpus. Out-of-vocabulary words are initialized randomly. The dimension of word embeddings is denoted as d. We define $X \in \mathbb{R}^{|V| \times d}$ as the set of word embeddings and the size of vocabulary is $|V|$.

[1] https://code.google.com/p/word2vec/.

When given a sentence s, we let $x_i \in \mathbb{R}^d$ be the d-dimensional word vector corresponding to the i-th word in the sentence. Hence, a sentence with the length of L is represented as a matrix: $s = (x_1; x_2; ...; x_L)$. In convolution layer, we use $WR^{(i)} \in \mathbb{R}^{k \times d}$ to represent the i-th convolution filter and $br^{(i)} \in \mathbb{R}$ to represent the bias term accordingly, where k is the context window size of the filter. Filter $WR^{(i)}$ will slide through the sentence s to get the latent features of sentence s. The sliding process can be represented as:

$$z_l^{(i)} = \sigma(WR^{(i)} * s_{l:l+k-1} + br^{(i)}), \tag{1}$$

where $z_l^{(i)}$ is the feature extracted by filter $WR^{(i)}$ from word x_l to word x_{l+k-1}. Hence, the latent features of the given sentence s are denoted as: $z^{(i)} = [z_1^{(i)}, ..., z_{L-k+1}^{(i)}]$. In order to extract keyword information of the sub-sentence, we apply the max-pooling operation to reserve the most prominent feature of filter $WR^{(i)}$ and denote it as:

$$z_{max}^{(i)} = max\{z^{(i)}\} = max\{z_1^{(i)}, ..., z_{L-k+1}^{(i)}\}. \tag{2}$$

We use multiple filters to extract multiple features. Therefore, the relation pattern of the given sub-sentence is represented as: $R_s = [z_{max}^{(1)}, ..., z_{max}^{(nr)}]$, where nr is the number of filters on RPE module.

3.3 The Module of Entity Semantic Extraction

The semantic properties of given entities contribute to reduce the impact of confusing relations. In this module, we focus on extracting the semantic properties of given entities based on their contextual words.

Word embeddings have been shown to preserve the semantic and syntactic information of words. But if we come across the unknown entity words, we still cannot obtain their semantic information from word embeddings. Fortunately, the properties of given entities can be reflected by their surrounding words. Different entities have different dependency on their contextual words. Some entities' property may be reflected by the former (next) one word, some may be reflected by the former (next) two words or more. Based on these motivations, we propose a mixture CNN to capture the semantic properties of entities as Fig. 3 shows.

We set entity word as the center, and select the sub-sentences with different scales around the entity as the entity's contexts. Extracting the entity's semantic properties is mining the semantics of these contexts. We still use CNN to extract the entities' contextual features. As Fig. 3 shows that $CNN \pm 1$ focuses on extracting the contextual semantic which is from word "early" to "of". $CNN \pm j$ mines the semantic information of context, which contains *2*j* surrounding words of entity "history". The architectures of CNNs we used here are the same as Sect. 3.2 described. We use $WE1_j^{(i)}$ to represent the i-th filter of $CNN \pm j$ on the ESE module for entity *e1* and $WE2_j^{(i)}$ to represent the i-th filter of $CNN \pm j$ on the ESE module for entity *e2*. Entity *e1*'s feature extracted by

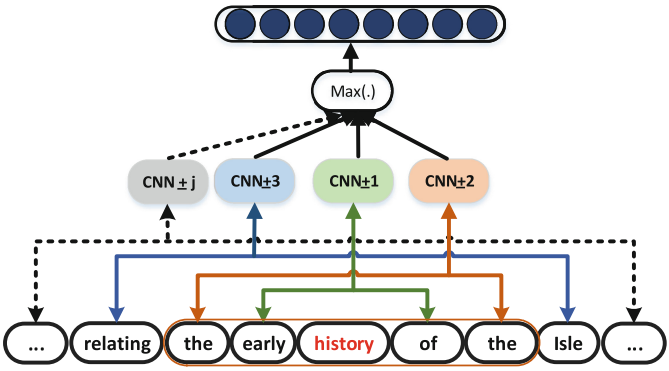

Fig. 3. The module of entity semantic extraction.

$WE1_j^{(i)}$ are denoted as $ze_j^{(i)}$. Hence, the j-th contextual information of entity $e1$ can be represented as $E1_j = [ze_j^{(1)}, ..., ze_j^{(ne)}]$, where ne is the number of filters on ESE module. Considering that different entities have different dependency on the contextual words, we apply a kind of max-pooling operation to merge the features extracted by $CNN \pm (1, 2...j)$. Namely,

$$E1_s = \begin{pmatrix} max(ze_1^{(1)} \ ... \ ze_j^{(1)}) \\ ... \quad ... \quad ... \\ max(ze_1^{(n)} \ ... \ ze_j^{(n)}) \end{pmatrix}. \tag{3}$$

3.4 Output Layer and Objective Function

After obtaining the semantic properties of given entities and relation pattern based on modules described in Sects. 3.2 and 3.3, we then merge these features by a concatenate manner which can be denoted as $f = [E1_s, R_s, E2_s]$. The output layer is the softmax classifier [15] with dropout:

$$y = W \cdot (f \circ r) + b, \tag{4}$$

$$p_i = \frac{exp(y_i)}{\sum_{j=1}^{m} exp(y_j)}, \tag{5}$$

where $W \in \mathbb{R}^{m \times (nr+2 \cdot ne)}$ is the weights between the merge layer and the layer of labels. m is the total number of relation classes. Symbol $\circ$ denotes the element-wise multiplication operator and $r \in \mathbb{R}^{(nr+2 \cdot ne)}$ is a binary mask vector drawn from bernoulli with probability ρ. Dropout guards against overfitting and makes the model more robust. In Formula 5, p_i means the probability that the merge features reflect the relation i.

The objective function of the method is to minimise the cross entropy errors between the distribution of predicted labels and the distribution of actual labels. It is defined as:

$$L = -\sum_{s \in S} \sum_{i=1}^{m} -log(P(y_i|s, \Theta)), \tag{6}$$

where S represents the sentences in training set and y_i is the correct class of the given sentence s. Θ is the parameters of the model, which can be concluded as: $\Theta = \{X, WR^{(i)}, br^{(i)}, WE1_j^{(i)}, be1_j^{(i)}, WE2_j^{(i)}, be2_j^{(i)}, W, b\}$.

The model is optimized by using stochastic gradient descent [16]. The gradients are obtained via backpropagation. Gradients are backpropagated only through the unmasked units in the layer with dropout. Besides, the learned parameters of weight, in the dropout layer, need to be scaled by ρ such that $W = \rho \cdot W$.

4 Experimental Setup

Dataset. To evaluate the performance of our method, we use SemEval-2010 Task 8 dataset [9] that is the widely used for relation classification. The dataset contains 8,000 sentences for training, and 2,717 sentences for testing. There are 9 directional relations and one additional "other" relation, which is used to represent the relation that does not belong to any of the nine main relations. The directional relations are "Cause-Effect(C-E)", "Component-Whole(C-W)", "Content-Container(C-C)", "Entity-Destination(E-D)", "Entity-Origin(E-O)", "Instrument-Agency(I-A)", "Member-Collection(M-C)", "Message-Topic(M-T)" and "Product-Producer(P-P)". Especially, "Cause-Effect(e1,e2)" and "Cause-Effect(e2,e1)" are different relations. "Cause-Effect(e1,e2)" means that e1 causes e2 and "Cause-Effect(e2,e1)" means e1 is caused by e2. Hence, there are 19 relation classes in total.

Metric. To compare the performance of different methods, we adopt the official metric, the macro-averaged F1 score defined by Hendrickx [9]. The metric computes the macro-averaged F1-scores for the nine actual relations (excluding other) and takes the directionality into consideration [4].

Baselines. The baselines we used are recent methods for the SemEval-2010 Task 8 and they can be mainly cast into two main categories: the handcrafted feature based methods and the neural network based methods.

The handcrafted feature based methods are proposed by Rink [1]. All of these methods use a considerable amount of resources (WordNet, and FrameNet, for example) then employ SVM [23] or MaxEnt [24] as the classifier. The results of handcrafted feature based methods are shown in the first five rows of Table 3.

Recently, neural network models have made significant progress in the task of relation classification. The neural network models are Convolutional Neural Network (CNN) based methods [3,4,8], Recursive Neural Network (RecNN) based methods [7] and Long Short Term Memory Network (LSTM) based methods [5].

- **CNN** [3] is the early work that exploits a convolutional deep neural network to extract lexical and sentence level features for the task of relation classification.
- **CR-CNN** [4] also applies CNN to classify relation classes. Instead of using softmax function on the top layer of CNN, it employs a pair-wise ranking strategy to reduce the effect of the confusing relation "Other".
- **depLCNN** [8] learns relation representations from shortest dependency paths through a convolution neural network. Besides, it also proposes a negative sampling strategy to improve the assignment of subjects and objects and can achieve the state-of-the-art results by using the external resources such as WordNet.
- **RNN** [7] introduces a recursive neural network model that learns compositional vector representations for sentences. Then it uses the sentences representations for the task of relation classification.
- **MV-RNN** [7] finds the path between the two entities in the constituent parse tree and learns the distributed representation of its highest node. It uses that node's vector as feature to classify the relationship.
- **SDP-LSTM** [5] leverages the shortest dependency path (SDP) between two entities; multichannel recurrent neural networks, with long short term memory (LSTM) units, pick up heterogeneous information along the SDP. It is the first to use LSTM-based recurrent neural networks for the relation classification task.
- **FCM** [6] decomposes the sentence into substructures and extracts features for each of them, forming substructure embeddings. These embeddings are combined by sum-pooling and inputed into a softmax classifier.

Hyper Parameter Settings. The hyper parameters used in these experiments are summarized in Table 2. On the ESE module, we set a series of CNNs to model entities' contextual information. The context window size of each CNN on ESE module is set to 5. If the length of input contextual sentence is less than 5, the context window size of this CNN is set to the length of the input.

Table 2. Hyper parameters of the mixture convolutional neural network (MixCNN)

Parameter symbol	Parameter description	Parameter value
d	Dimension of word embedding	300
nr	The filter number of CNN on RPE module	300
ne	The filter number of CNN on ESE module	1000
k	Context window size of RPE module	20
j	The number of CNNs on ESE module	4
ρ	The ratio of dropout in merged layer	0.3

5 Results

5.1 Comparison with the Baselines on Relation Classification

We compare our method with the baselines which are recently published for the SemEval-2010 Task 8. In order to achieve state-of-the-art results, some approaches need to add external information such as: Word-Net, FrameNet or other NLP resources, which is actually an unfair comparison. Because different external resources have different effect on improving the predicted results. Besides, methods using external information have limitations. For example, if a method uses WordNet, it only suits for the task in English. To better illustrate the effectiveness of our method, we do not use any external information except word embedding in this experiment. We report the results of different methods as Table 3 shows.

In Table 3, only using word embedding as input features, our method achieves F1 of 84.8 %, which is the best results comparing with other methods. It shows that joint learning of entities' semantic properties and relation keywords is good for the task of relation classification. [3] is the early work that using CNN to classify relation. Although CNN [3] can extract sentence level features, it cannot achieve good results when only using word embedding features. Santos [4] also employs a kind of CNN method, called CR-CNN, to do the task by proposing a new pairwise ranking loss function. It can achieve the result of 84.1 %. The pairwise ranking loss function can reduce the impact of "Other" class. If it uses log-loss instead of the task-specific pairwise ranking loss function, the *F1* value is only 82.5 % which also has two percentage points worse than our method. Although our method uses the softmax, it can be also superior to CR-CNN with the pairwise ranking loss function. depLCNN [8] combines the dependency path and CNN to represent the sentence and can achieve the results of 81.3 %. Apart from the CNN methods, there are many sequential neural networks [5,7], which achieve results from 74.8 % to 82.4 %. [6] is a factor based compositional embedding model that only achieves the *F1* of 80.4 %.

We also compare our method with these baselines by adding external resources as Table 3 shows. Although we do not use any external information except word embeddings, our method still defeats most baselines which use lexical resources or NLP tools. If depLCNN only uses a negative sampling strategy to increase the number of training samples, our method can still has *+0.8 %* improvement. Besides, if depLCNN uses WordNet and negative sampling strategy simultaneously, we can also get comparable results to theirs under the circumstance that our training set is the half of theirs and without using WordNet.

5.2 The Effectiveness for Extracting Entity Semantic

In this paper we are not only focusing on achieving the state-of-the-art results on relation classification without using any external information, but also providing an effective manner to extract the semantic properties of given entities. In order

Table 3. Comparison of methods with adding different external resources. The external resources can be WordNet or other information obtained by NLP tools. Different resources have different effect on improving the predicted results. To better illustrate the effectiveness of our method, we do not use any external information except word embedding in this experiment.

Method	External resources	F1(%)
SVM [1]	POS, stemming, syntactic patterns	60.1
SVM [1]	word pair, words in between	72.5
SVM [1]	POS, stemming, syntactic patterns, WordNet	74.8
MaxEnt [1]	WordNet, FrameNet, Google n-grams, morphological	77.6
SVM [1]	WordNet, FrameNet, Google n-grams, morphological	82.2
RNN [7]	—-	74.8
RNN [7]	POS, NER, WordNet	77.6
MVRNN [7]	—-	79.1
MVRNN [7]	POS, NER, WordNet	82.4
FCM [6]	—-	80.6
FCM [6]	Dependency parse, NER	83.0
SDP-LSTM [5]	—-	82.4
SDP-LSTM [5]	POS, WordNet, Grammar relation	83.7
CNN [3]	—-	69.7
CNN [3]	WordNet	82.7
depLCNN [8]	—-	81.3
depLCNN [8]	Negative sampling	84.0
depLCNN [8]	WordNet	83.7
depLCNN [8]	WordNet, Negative sampling	**85.6**
CNN + softmax [4]	—-	82.5
CNN + CR [4]	—-	84.1
MixCNN + CNN	—-	**84.8**

to further illustrate the effectiveness of ESE module on representing the semantic properties of given entities, we also conduct cluster experiments.

We use $ESE(e)$ to represent the semantic embedding of entity e that is extracted by module ESE. Hence the semantic relation between an entity pair $(e1, e2)$ can be denoted as $R_{ese}(e1, e2) = ESE(e1) - ESE(e2)$.

Word embeddings have been empirically shown to preserve semantic relation between words [14]. For example, $v(king) - v(queen) \approx v(man) - v(woman)$. $v(w)$ is the word embedding of word w. We use $R_v(e1, e2) = v(e1) - v(e2)$ to represent semantic relation between $e1$ and $e2$, which is initialized by word2vec. Here, we use $R_v(e1, e2)$ as our baseline.

Given the datasets, we obtain the relation embeddings of each entity pair: $R^i_{ese}(e1, e2)$ and $R^i_v(e1, e2)$. Then we employ the K-means algorithm [18] to cluster relation embeddings produced by the above manners. The clustering performance is evaluated by comparing the clustering results of texts with the relation labels provided by the datasets. Two metrics, the accuracy (ACC) [20] and the normalized mutual information (NMI) metrics [19], are used to measure the clustering performance [22]. Given a text x_i, let c_i be the predicted cluster label and y_i be the true label provided by corpus. Then the accuracy is defined as:

$$ACC = \frac{\sum_{i=1}^{n} \delta(y_i, c_i)}{n}, \tag{7}$$

where n is the size of dataset and $\delta(x, y)$ is the indicator function that equals one if $x = y$ and equals zero otherwise. Normalized mutual information is a popular metric used for evaluating clustering tasks. It is defined as:

$$NMI(\mathbf{Y},\mathbf{C}) = \frac{MI(\mathbf{Y},\mathbf{C})}{\sqrt{H(\mathbf{Y})H(\mathbf{C})}}, \tag{8}$$

where $MI(\mathbf{Y},\mathbf{C})$ is the mutual information between the predicted label set $\mathbf{Y}$ and the target label set $\mathbf{C}$. $H(.)$ is the entropy and $\sqrt{H(\mathbf{Y})H(\mathbf{C})}$ is used for normalizing the mutual information [22].

We run 100 times for each experiment and obtain the final results as Table 4 shows. The experimental results show that R_{ese} significantly better than R_v on both the accuracy (ACC) and the normalized mutual information (NMI) metrics. Although word embeddings can preserve the semantic and syntactic information of words, when we come across the unknown entity words, word embedings can do nothing. Besides, word embeddings contain much complex semantic information, so the semantic relation of word embedding is not obvious. ESE extracts the semantic properties of given entities by using their contextual information, which can solve the problem of unknown entity words. Furthermore, ESE focuses on mining relation properties of entities instead of modeling the complex semantic and syntactic information. Therefore, the R_{ese} significantly better than R_v.

Table 4. Comparison of ACC and NMI of K-means cluster algorithm based on different relation representations.

Dataset	Train		Test	
Relation Embedding	R_v	R_{ese}	R_v	R_{ese}
ACC(%)	26.98 ± 1.11	76.17 ± 5.13	23.99 ± 0.99	60.12 ± 2.95
NMI(%)	22.19 ± 0.81	84.72 ± 1.96	20.58 ± 0.76	61.16 ± 0.91

We also visualize the clustering results by using t-SNE [21] as Fig. 4 shows. In the embedding space produced by ESE, the entity pairs with same relation are

more close to each other and the entity pairs with different relation are far from each other. On the contrary, there is no such obvious rule in the word embedding space produced by word2vec. The results further illustrate the effectiveness of ESE module on representing the semantic properties of given entities.

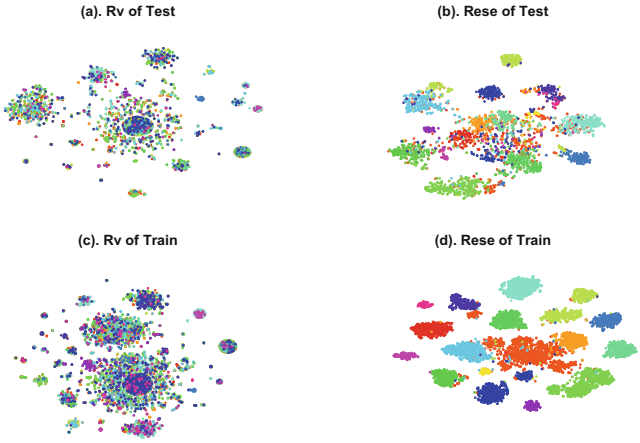

Fig. 4. The t-SNE visualization of the relation embeddings. Figure (a) and (c) are the relation embeddings produced by word2vec on training set and testing set. Figure (b) and (d) are produced by ESE module.

6 Analysis and Discussion

6.1 Module Analysis

In order to extract the relation pattern and obtain the semantic properties of given entities, we set two modules: RSE described in Sect. 3.2 and ESE in Sect. 3.3. In this section, we focus on analyzing the properties of these two modules.

At first, we allow each module with different configurations to perform the task of relation classification. We adopt a CNN architecture [12] called RPE to model the sub-sentence, which is the words between the given entities, instead of the whole sentence. We also compare the results of full sentence configuration which is marked as RPE1. Besides, we propose ESE module to extract the semantic properties of given entities based on their contextual words. In order to better verify the effectiveness of ESE module, we directly use entity word embedding to represent entity information. The embedding of unknown entity word is initialized randomly. We mark this configuration as ESE1. In addition to testing the effects of each module alone, we also test their various combinations. In this paper, our method is the combination of RPE and ESE.

From Table 5, we know that RPE and ESE can also achieve comparable results of *F1* when compared with most of the baselines. Besides, when compared

Table 5. Comparison of the modules on the task of relation classification.

Methods	Input Text	*Prec.(%)*	*Rec.(%)*	*F1(%)*
RPE	sub-sentence	80.9	84.6	82.6
RPE1	full-sentence	72.4	74.5	73.3
ESE	entity-context	81.8	84.6	83.1
ESE1	entities	65.4	58.5	61.5
RPE1+ESE1	full-sentence, entities	70.9	75.6	73.1
RPE1+ESE	full-sentence, entity-context	81.3	84.2	82.7
RPE+ESE1	sub-sentence, entities	80.6	82.8	81.6
RPE+ESE	sub-sentence, entity-context	**83.1**	**86.6**	**84.8**

with RPE1 that extracts the relation pattern from full sentence, RPE achieves a +10 % improvement. It matches our observations and Santos' [4] analysis that most relation pattern can be reflected by the sub-sentence between the given two entities. When compared with ESE1, ESE achieves a +26 % improvement. These results verify the rationality of our motivations and the effectiveness of the proposed modules. Besides, merging ESE and RPE can bring about 2 points of improvement in *F1* value, which shows the complementarity of the two modules as well as the necessity of module integration.

6.2 Error Analysis

We conduct extensive qualitative and quantitative analysis of errors to better understand our method in terms of learning and predicting quality. We visualized the model's predicted results as Fig. 5 shows.

The diagonal region indicates the correct prediction results and the other regions reflect the distribution of error samples. The highlighted diagonal region means that our method can perform well on each relation class. However, from Fig. 5, we also can see that the distribution of predicted relation is relatively dispersed on the last column of "Other". Besides, most of the specific relation classes can be predicted as the "Other", which reflected from the last row shows in Fig. 5. The class of "Other" is a kind of confusing and heterogeneous class. It contains many different kinds of relation classes. Although our method can reduce the impact of confusing classes, it still need further improvement for the class of "Other". Apart from the class "Other", the class "I-A(e1,e2)" perform worse than the other 17 classes. Based on our observations, we find there are many samples in the class "I-A(e1,e2)" have the property that the given two entities are usually close to each other at the beginning of a sentence. For examples: "Elevator(e1) operator(e2) is a meditation on the..." and "Camera(e1) operator(e2) is that person ...". Because, there is no indicative words between two entities and there are few contextual words around entities. Our method is inadequate to deal with this case.

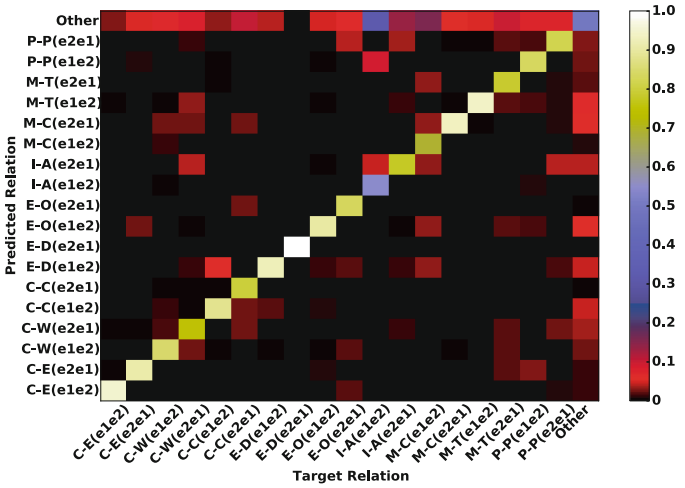

Fig. 5. The distribution of the predicted results for each relation class. The horizontal axis is the target relation and each target relation corresponds to a column of predicted relations. Point (X, Y) means the ratio that the target relation is X and the predicted relation is Y. The sum of each column value equal to 1.

7 Conclusion

In this paper, we propose a mixture convolutional neural network, which is an unified model by jointly learning entities' semantic properties and relation pattern, to fix the task of relation classification. It can achieve the state-of-the-art results on the SemEval-2010 Task 8 dataset without using any external information. Besides, we also conduct experiments to show that the entity embedding generated by our approach can reflect the relation properties of given entities.

Although our method can help to reduce the impact of the confusing relation, it still need further improvement for the class of "Other". In the future, we will focus on solving the problem of the special class "Other" and test our method on more related datasets.

Acknowledgements. This work is also supported by the National High Technology Research and Development Program of China (863 Program) (Grant No. 2015AA015402), the Hundred Talents Program of Chinese Academy of Sciences (No. Y3S4011D31), the NSFC project (No. 61501463) and National Natural Science Foundation (Grant No. 71402178).

References

1. Rink, B., et al.: UTD: Classifying semantic relations by combining lexical and semantic resources. In: 5th SE, pp. 256–259 (2010)
2. Kambhatla, N.: Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In: 43th ACL, pp. 22–26 (2004)

3. Zeng, D., et al.: Relation classification via convolutional deep neural network. In: 25th COLING, pp. 2335–2344 (2014)
4. dos Santos, C., Nogueira, et al.: Classifying relations by ranking with convolutional neural networks. In: 53th ACL, pp. 626–634 (2015)
5. Xu, Y., et al.: Classifying relations via long short term memory networks along shortest dependency paths. In: EMNLP (2015)
6. Yu, M., et al.: Factor-based compositional embedding models. In: NIPS Workshop on Learning Semantics (2014)
7. Socher, R., et al.: Semantic compositionality through recursive matrix-vector spaces. In: EMNLP, pp. 1201–1211 (2012)
8. Xu, K., et al.: Semantic relation classification via convolutional neural networks with simple negative sampling. In: EMNLP (2015)
9. Hendrickx, I., et al.: Semeval-2010 task 8: multi-way classification of semantic relations between pairs of nominals. In: SE, pp. 94–99 (2009)
10. Sun, L., Han, X.: A Feature-Enriched Tree Kernel for Relation Extraction. In: the 52th ACL, pp. 61–67 (2014)
11. Blunsom, P., et al.: A convolutional neural network for modelling sentences. In: 52th ACL, (2014)
12. Kim, Y.: Convolutional neural networks for sentence classification. In: EMNLP, pp. 1746–1751 (2014)
13. Collobert, R., et al.: Natural language processing (almost) from scratch. In: JMLR, pp. 2493–2537 (2011)
14. Mikolov, T., et al.: Distributed representations of words and phrases and their compositionality. In: NIPS, pp. 3111–3119 (2013)
15. Duan, K., et al.: Multi-category classification by soft-max combination of binary classifiers. J. Multiple Classifier Syst., 125–134 (2003)
16. Bottou, L.: Stochastic gradient learning in neural networks. J. Neuro-Nımes **9** (1991)
17. Fu, R., et al.: Learning semantic hierarchies via word embeddings. In: 52th ACL, pp. 1199–1209 (2014)
18. Wagstaff, K., et al.: Constrained k-means clustering with background knowledge. In: 18th ICML, pp. 577–584 (2001)
19. Chen, W.-Y., et al.: Parallel spectral clustering in distributed systems. J. TPAMI, 568–586 (2011)
20. Cai, D., et al.: Document clustering using locality preserving indexing. IEEE Trans. J. Knowl. Data Eng., 1624–1637 (2005)
21. Van der Maaten, L., et al.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**, 2579–2605 (2008)
22. Jiaming, X., Peng, W., et al.: Short text clustering via convolutional neural networks. In: The NAACL, pp. 62–69 (2015)
23. Hearst, M.A., Dumais, et al.: Support vector machines. In: IEEE Intelligent Systems and their Applications, pp. 18–28 (1998)
24. Phillips, S.J., et al.: Maximum entropy modeling of species geographic distributions. In: Ecological modelling, pp. 231–259 (2006)