Knowledge-Based Systems 000 (2016) 1–12

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A neural network framework for relation extraction: Learning entity semantic and relation pattern

Suncong Zheng[a], Jiaming Xu[a], Peng Zhou[a], Hongyun Bao[a,*], Zhenyu Qi[a], Bo Xu[a,b]

[a] Institute of Automation, Chinese Academy of Sciences (CAS), China
[b] Center for Excellence in Brain Science and Intelligence Technology, CAS, China

ARTICLE INFO

Article history:
Received 1 April 2016
Revised 22 September 2016
Accepted 23 September 2016
Available online xxx

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

ABSTRACT

Relation extraction is to identify the relationship of two given entities in the text. It is an important step in the task of knowledge extraction. Most conventional methods for the task of relation extraction focus on designing effective handcrafted features or learning a semantic representation of the whole sentence. Sentences with the same relationship always share the similar expressions. Besides, the semantic properties of given entities can also help to distinguish some confusing relations. Based on the above observations, we propose a neural network based framework for relation classification. It can simultaneously learn the relation pattern's information and the semantic properties of given entities. In this framework, we explore two specific models: the CNN-based model and LSTM-based model. We conduct experiments on two public datasets: the SemEval-2010 Task8 dataset and the ACE05 dataset. The proposed method achieves the state-of-the-art result without using any external information. Additionally, the experimental results also show that our approach can represent the semantic relationship of the given entities effectively.

1. Introduction

Relation extraction is to identify semantic relation of the entity pairs in one sentence. 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 classification focus on designing effective handcrafted features to obtain better classification performance [1–4]. These handcrafted features are extracted by analyzing the text and using different natural language processing (NLP) tools. However, they decompose the task into several pipelined components. For example: firstly they use POS (or Parser) tool to extract features, then classify relations based on these features. Errors in POS (or Parser) procedure will be propagated to the relation classification, which is the error propagation. In order to reduce the manual work in feature extraction, recently, deep neural networks [5–8] have been applied to obtain effective relation features of sentences. Although these models can learn related features from given sentences without complicated feature engineering work, most of the current deep learning approaches mainly focus on learning the semantic representation of the whole sentence.

Properties of the task. Through the observation, we find that sentences with the same relation label always share the similar expressions. We call the similar expression as relation pattern in this paper. We randomly select some instances from the SemEval-2010 Task8 dataset [3] as Table 1 shows. For examples: 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" or "to". 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" or "was generated via". These findings and the previous work's analysis [9] both show that most relation patterns can be reflected by the keywords information or the syntactical information in a sentence, especially the sub-sentence between the given entities. Therefore, when compared with learning a semantic embedding of the whole sentence, extracting the sub-sentence information between given entities can better reflect the relation pattern in the sentence. However, it is not enough to distinguish the confusing relation category by only using the relation pattern 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. Thus, making good

* Corresponding author.
E-mail address: hongyun.bao@ia.ac.cn (H. Bao).

http://dx.doi.org/10.1016/j.knosys.2016.09.019
0950-7051/

Table 1
Instances in the SemEval-2010 Task8 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.
3) The [*singer*]$_{e1}$ **arrived to** the outdoor [*stage*]$_{e2}$ for rehearsal.
Cause-Effect(e2,e1):
4) Plantar [*warts*]$_{e1}$ **are caused by** a [*virus*]$_{e2}$ that infects the layer of skin on the soles of feet.
5) A wind speed associated with the [*devastation*]$_{e1}$ **caused by** the [*tornado*]$_{e2}$.
6) When the [*force*]$_{e1}$ **was generated via** the [*joystick*]$_{e2}$, the reproduced force matched the original force much more accurately.
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*]$_{e2}$.

use of given entities' semantic properties can also help to distinguish the confusing relationship.

Motivations of our method. Based on the above analysis, we propose an unified framework for relation classification by jointly learning relation pattern and entity semantics. When extracting relation pattern, we propose two kinds of neural network based methods, Convolutional neural networks (CNN) [10] and Long Short-Term Memory (LSTM) [11]. Convolutional neural networks (CNN) [10,12–14] have achieved great success in sentence's semantic representation. It is able to preserve sequence information and extract the keyword information in a sentence. Most relation patterns can be reflected by some keywords in a sentence, especially the words between the given entities. Therefore, we adopt a CNN architecture to model the sub-sentence between given entities instead of modeling the whole sentence. In addition to CNN, we also use LSTM to extract relation pattern. The LSTM model and its different variants [8,15–19] have achieved impressive performance in text analysis. It is useful in capturing long-range dependencies of sequences and has shown powerful results on extracting syntactical information of sentences [17–19]. Since the similar expressions of sentences can be also reflected by the syntactical information of sentences, we use LSTM to extract relation pattern.

Besides, we propose a novel mixture CNN model to extract the semantic properties of given entities. The semantic properties of given entities can be reflected by their contextual words. Some entities' properties may be reflected by one contextual word, some may be reflected by two or more contextual words. Extracting the entity's semantic properties is to mine the semantic information of these contextual sub-sentences. Hence, we apply the operation of mixture convolution to extract the entities' different contextual features. The mixture CNN model sets the entity word as the center, and selects different sub-sentences around the entity as the entity's contexts, then uses 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 method we proposed, in this paper, is a neural network based framework for relation classification, which can simultaneously learn the semantic properties of entities and relation pattern. Based on this framework, we set up two specific models from different perspectives: the CNN-based model and the LSTM-based model. Each model contains two modules, the entity semantic extraction module (ESE) and the relation pattern extraction module (RPE). Different modules focus on extracting different information, and all information is merged in the output layer to fix the task of relation classification.

Contributions. The main contributions of this paper can be summarized as follows:

1. We propose a neural network based framework for the task of relation classification, which can simultaneously learn the semantic properties of entities and sentence's relation pattern.
2. Based on this framework, we also explore two specific models: the CNN-based model, which focuses on extracting keyword information, and the LSTM-based model, which is to learn sentence's syntactical information.
3. Besides, we also conduct experiments to analyze the entity embedding produced by our method. Our approach can represent semantic relationship of given entities effectively, when compared with word2vec [20].

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 [1,2,4], neural network based methods [5–9] and the other valuable methods [6,21].

The handcrafted feature based methods focus on using different natural language processing (NLP) tools and knowledge resources to obtain effective handcrafted features. Then, they always use some statistical classifier such as Support Vector Machines (SVM) [22] or Maximum Entropy (MaxEnt) [23] to get the right relation class based on the handcrafted features. Kambhatla [1] employs Maximum Entropy model to combine diverse lexical, syntactic and semantic features derived from the text. It is the early work for relation classification. The features they used are not comprehensive. Rink et al. [4] designs 16 kinds of features that are extracted by using many supervised NLP toolkits and resources including POS, Word-Net, dependency parse, etc. It can get the best result at SemEval-2010 Task 8 when compared with other handcrafted features based methods. However, it relies heavily on other NLP tools and it also requires a lot of work to design and extract features.

In recent years, deep neural models have made significant progress [5–9,24]. Neural network models can 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) [7,9,25] and sequential neural networks such as Recurrent Neural Networks (RNN) [26], Recursive Neural Networks (RecNN) [5,27] and Long Short Term Memory Networks (LSTM) [8,28].

Zeng et al. [7] 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. dos Santos et al. [9] and Xu et al. [25] also apply convolutional neural network to classify relation classes. dos Santos et al. [9] uses a pair-wise ranking method instead of softmax function on the top of CNN to reduce the effect of the confusing relation "Other". Xu et al. [25] uses a convolution neural network through the shortest dependency paths to learn more robust relation representations. Besides, it also propose a negative sampling strategy to improve the assignment of subjects and objects.

Please cite this article as: S. Zheng et al., A neural network framework for relation extraction: Learning entity semantic and relation pattern, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.09.019

Apart from the CNN methods, there are many sequential neural networks. Socher et al. [5] is the first work that introduces a kinds of recursive neural network (RNN) model to classify relation. The RNN model has the shortage that it cannot capture the long-distance relationships of words. For effective information propagation and integration, [8] leverages long short term memory (LSTM) units during recurrent propagation. Zhang et al. [28] further proposes a bidirectional long short-term memory network (BLSTM) to model the sentence with complete, sequential information about all words before and after it. However, the experimental results show that BLSTM [28] based method only with word embeddings as input features is sufficient to achieve state-of-the-art performance.

There also exists other valuable methods such as the kernel-based methods [21,29] and compositional model [6]. Nguyen et al. [21] explores the use of innovative kernels based on syntactic and semantic structures. Sun and Han [29] proposes a new tree kernel, called feature-enriched tree kernel (FTK) for relation extraction. The compositional model FCM [6] learns representations for the substructures of an annotated sentence. Compared to existing compositional models, FCM can easily handle arbitrary types of input and global information for composition.

The methods used in this paper are based on neural network models: Convolutional neural networks (CNN) and Long Short-Term Memory (LSTM). CNN is originally invented for computer vision [30]. It utilizes layers with convolving filters to extract local features. In recent years, CNNs have been successfully applied to different NLP tasks and have also shown the effectiveness on extracting sentence semantic and keywords information [9,10,12,13,13]. Long-Short Term Memory (LSTM) model is a specific kind of recurrent neural networks (RNNs). LSTM replaces the hidden vector of a recurrent neural network with memory blocks which are equipped with gates. It can keep long term memory by training proper gating weights [31,32]. LSTM have also shown powerful capacity on many NLP tasks such as machine translation [19], sentence representation [17,33] and relation extraction [8].

In this paper, we propose a neural network based framework by joint learning the entity semantic properties and relation pattern for the task of relation classification. It can learn related features from given sentences without complicated feature engineering work, when compared with handcrafted feature based methods. Besides, different from the other neural network based methods that focus on learning the semantic representation of the whole sentence, our method pays attention to model the relation pattern and takes full advantage of the given entities' semantic properties. It can learn semantic properties of given entities and sentence's relation pattern effectively.

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 an unified model by jointly learning entities' semantic properties and relation patterns. Based on this framework, we set up two specific models, the CNN-based model called MixCNN+CNN and the LSTM-based model called MixCNN+LSTM. MixCNN+CNN and MixCNN+LSTM share the same entity semantic extraction module (ESE). The difference is about relation pattern extraction module (RPE). MixCNN+CNN uses a kind of CNN to extract relation pattern and MixCNN+LSTM uses a LSTM model to extract 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.

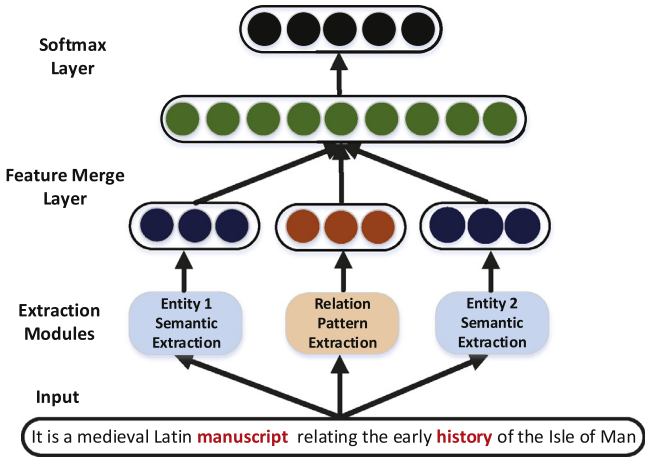

Fig. 1. The framework of our method.

3.1. Overview of our method

The framework of method 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 semantic or syntactical information between the two given entities. Especially, in the relation pattern extraction module (RPE), we set up two kinds of models to extract the relation pattern from different perspectives. The CNN-based model focuses on extracting keyword information and the LSTM-based model focuses on representing sentence's syntactical information. 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 classify relations. In what follows, we describe these modules in detail.

3.2. The module of relation pattern extraction

In this section, we focus on extracting relation pattern by using two kinds of neural network based methods. Firstly, we introduce the CNN-based model in Section 3.2.1, then we present the LSTM-based model in Section 3.2.2.

3.2.1. Relation pattern extraction based on CNN

Based on our observations and dos Santos et al.'s [9] analysis, 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) [10,12,13] is able to preserve the sequence information and extract the keyword information in a sentence. Xu et al. [25], Zeng et al. [7], and dos Santos et al. [9] also validate the effectiveness of CNN to extract the related keyword information. Therefore, in order to extract the keyword information which can reflect relation patterns, we adopt the CNN architecture [10] to model the sub-sentence between the given entities instead of using the whole sentence as Fig. 2 shows.

Firstly, each word is represented by a word embedding. In our experiments, we employ the word2vec[1] [20] to produce the word embeddings based on Wikipedia corpus. Out-of-vocabulary words are initialized randomly. The dimension of word embeddings is de-

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

Please cite this article as: S. Zheng et al., A neural network framework for relation extraction: Learning entity semantic and relation pattern, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.09.019

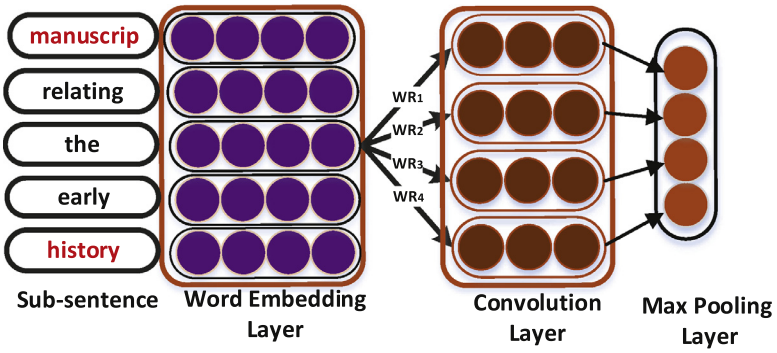

Fig. 2. Relation pattern extraction based on CNN.

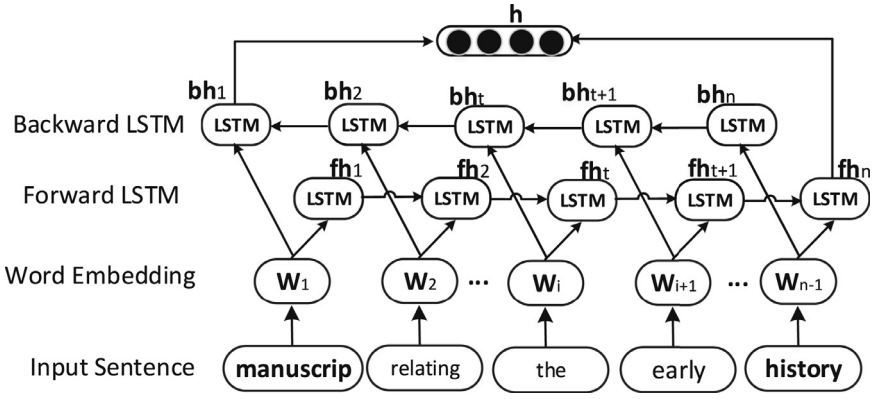

Fig. 3. The framework of bi-directional LSTM.

noted as d. We define $X \in \mathbb{R}^{|V|\times d}$ as the set of word embeddings and the size of vocabulary is $|V|$.

When given a sentence s, we let $x_i \in \mathbb{R}^d$ be the d-dimensional word vector corresponding to the ith word in the sentence. Hence, a sentence with the length of L is represented as a matrix: $s = (x_1; x_2; \ldots; x_L)$. In convolution layer, we use $WR^{(i)} \in \mathbb{R}^{k\times d}$ to represent the ith 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:

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

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: $z^{(i)} = [z_1^{(i)}, \ldots, z_{L-k+1}^{(i)}]$. 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:

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

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

3.2.2. Relation pattern extraction based on LSTM

LSTM is useful in capturing long distance relationship in different fields and it also shows powerful ability on extracting syntactical information of sentences [17–19]. Besides, the sentences with same relation label always share the similar expressions, which can be reflected by the syntactical information of sentences. With this motivations, we adopt LSTM to get the relation pattern information. Unlike previous work, we use a bi-directional long short-term memory networks (BLSTM) to represent the sub-sentence between the given entities. Because it is mostly sufficient to use only the words between the two given entities [8,9]. Besides, the relationship between the given entities has the directional properties. For example, "Cause-Effect(e1,e2)" and "Cause-Effect(e2,e1)" are different relations.

The bi-directional LSTM model we used to extract relation pattern are shown in Fig. 3. It has five kinds of layers: input layer, word embedding layer, forward hidden layer, backward hidden layer and the concatenate layer. The input layer encodes words into 1-hot representation from the input sentence. Then the word embedding layer converts the word with 1-hot representation to an embedding vector. Hence, a sequence of words can be represented as $X = \{x_1, x_2, \ldots x_{t-1}, x_t \ldots x_L\}$, where $x_i \in \mathbb{R}^d$ is the d-dimensional word vector corresponding to the ith word in the sentence and L is the length of the given sentence. After word embedding layer, there are two parallel LSTM layers: forward hidden layer and backward hidden layer. At each time-step t, the forward hidden layer will compute a hidden representation fh_t of the sub-sentence that contains words from x_1 to x_t. The backward hidden layer will compute a hidden representation bh_t of the sub-sentence that contains words from x_L to x_t. At last, we merge the hidden representations of forward and backward layers by concatenating bh_1 and fh_n. Hence, the representation of relation pattern can be denoted as $R_s = [bh_1, fh_n]$.

The LSTM architecture consists of a set of recurrently connected subnets, known as memory blocks. Each time-step in forward hidden layer and backward hidden layer is a LSTM memory block. A block contains one or more self-connected memory cells and three multiplicative units: the input, output and forget gates. These gates provide continuous analogues of write, read and reset operations for the cells [32]. Fig. 4 provides an illustration of a LSTM memory block with a single cell. Each memory block takes a previous

Please cite this article as: S. Zheng et al., A neural network framework for relation extraction: Learning entity semantic and relation pattern, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.09.019

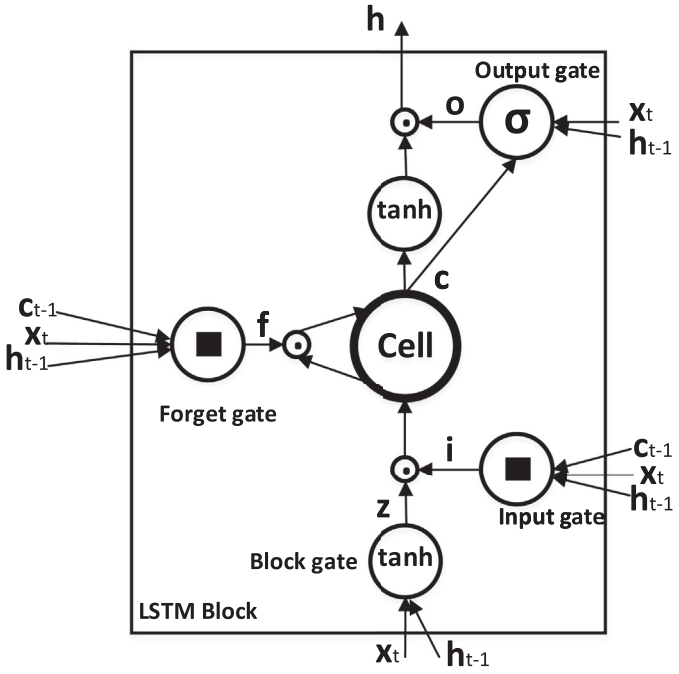

Fig. 4. LSTM memory block with one cell.

hidden vector h_{t-1} and a current input word embedding x_t then computes current hidden vector h_t, which can shortly defined as: $fh_t = LSTM(fh_{t-1}, x_t)$ and $bh_t = LSTM(bh_{t+1}, x_t)$. The detail operation of a LSTM memory block can be defined as follows:

$$i_t = \delta(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i), \tag{3}$$

$$f_t = \delta(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f), \tag{4}$$

$$z_t = tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c), \tag{5}$$

$$c_t = f_t c_{t-1} + i_t z_t, \tag{6}$$

$$o_t = \delta(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o), \tag{7}$$

$$h_t = o_t tanh(c_t), \tag{8}$$

where i, f and o are the input gate, forget gate and output gate respectively, b is the bias term, c is the cell memory and W is the parameters. In forward hidden layer, fh_t corresponds to h_t and fh_{t-1} corresponds to h_{t-1}. In backward hidden layer, bh_t corresponds to h_t and bh_{t+1} corresponds to h_{t-1}.

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 CNNs (MixCNN) to capture the semantic properties of entities as Fig. 5 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 also use CNN to extract the entities' contextual features. As Fig. 5 shows that $CNN \pm 1$ focuses on extracting the contextual semantic which is from word "early" to word "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 Section 3.2 described. We use $WE1_j^{(i)}$ to represent the ith filter of $CNN \pm j$ on the ESE module for entity *e1* and $WE2_j^{(i)}$ to represent the ith filter of $CNN \pm j$ on the ESE module for entity *e2*. Entity *e1*'s feature extracted by $WE1_j^{(i)}$ are denoted as $ze_j^{(i)}$. Hence, the jth contextual information of entity *e1* can be represented as $E1_j = [ze_j^{(1)}, \ldots, ze_j^{(ne)}]$, 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 $CNN \pm (1, 2 \ldots j)$. Namely,

$$E1_s = \begin{pmatrix} max(ze_1^{(1)}) & \ldots & ze_j^{(1)}) \\ \ldots & \ldots & \ldots \\ max(ze_1^{(n)}) & \ldots & ze_j^{(n)}) \end{pmatrix}. \tag{9}$$

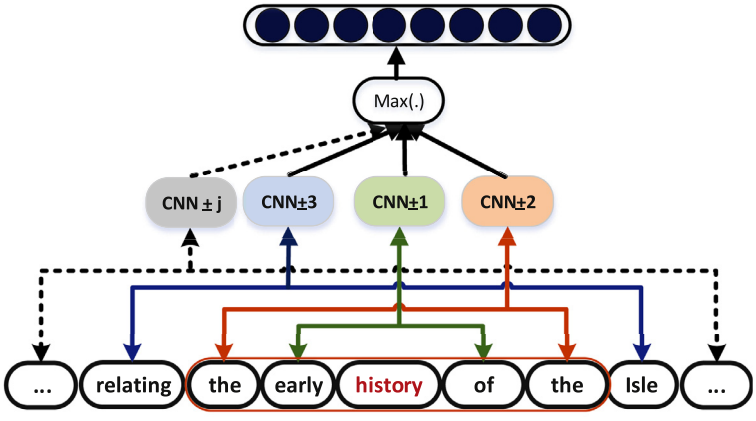

Fig. 5. The module of entity semantic extraction based on mixture CNN (MixCNN).

Please cite this article as: S. Zheng et al., A neural network framework for relation extraction: Learning entity semantic and relation pattern, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.09.019

3.4. Output layer and objective function

After obtaining the semantic properties of given entities and relation pattern, 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 [34] with dropout [35]:

$$y = W \cdot (f \circ r) + b, \tag{10}$$

$$p_i = \frac{exp(y_i)}{\sum_{j=1}^{m} exp(y_j)}, \tag{11}$$

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 11, p_i means the probability that the merge features reflect the relation i.

The objective 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:

$$L = -\sum_{s\in S}\sum_{i=1}^{m} -log(P(y_i|s,\Theta)), \tag{12}$$

where S is the sentences in training set and y_i is the correct class of the given sentence s. Θ is the parameters of the model. The parameters of MixCNN+CNN can be concluded as: $\Theta_{cnn} = \{X, WR^{(i)}, br^{(i)}, WE1_j^{(i)}, be1_j^{(i)}, WE2_j^{(i)}, be2_j^{(i)}, W, b\}$. Besides, the parameters of MixCNN+LSTM are $\Theta_{lstm} = \{X, WF_{(xi)}, WF_{(hi)}, bf_{(i)}, WF_{(ci)}, WF_{(xf)}, WF_{(cf)}, WF_{(hf)}, bf_{(f)}, WF_{(xc)}, WF_{(hc)}, bf_{(c)}, WF_{(xo)}, WF_{(co)}, WF_{(ho)}, bf_{(o)}, WB_{(xi)}, WB_{(hi)}, WB_{(ci)}, bb_{(i)}, WB_{(xf)}, WB_{(cf)}, bb_{(f)}, bb_{(o)}, bb_{(c)}, WB_{(hc)}, WB_{(xc)}, WB_{(xo)}, WB_{(co)}, WB_{(ho)}, WB_{(hf)}, WE1_j^{(i)}, be1_j^{(i)}, WE2_j^{(i)}, be2_j^{(i)}, W, b\}$.

The model is optimized by using stochastic gradient descent [36]. 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

4.1. Datasets

The most widely used dataset for relation classification is SemEval-2010 Task 8 dataset [3]. Hence, we use SemEval-2010 Task 8 dataset [3] to evaluate the performance of our method and analyze the property of model. Besides, in order to further illustrate the effectiveness of our method, we also conduct an auxiliary comparative experiments on a more sophisticated and difficult dataset: ACE05 (Automatic Content Extraction program).[2]

SemEval-2010 Task 8 dataset contains 8000 sentences for training, and 2717 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. The "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.

ACE05 is the dataset for entity mention extraction and relation extraction tasks. In this paper, we only use it to verify the effectiveness of our method on the relation extraction task. It has 6 coarse-grained relation types: "Person-Social", "Agent-Artifact", "GPE-Affiliation", "Part-Whole", "PHYS" and "Employment-Organization". The opposite directions of same relation are also considered to be two classes. We also add a non-relation classes, so there are 13 relation classes in total. The sentences in ACE05 are more complex than SemEval-2010 and the given entities can be pronoun word or other irregular words, which makes the task more difficult. The training dataset contains 60,152 sentences, and the size of testing dataset is 12,534.

4.2. Metric

To compare the performance of different methods, we adopt the official metric, the macro-averaged F1 score defined by Hendrickx et al. [3]. The metric computes the macro-averaged F1-scores for the actual relations (excluding other) and takes the directionality into consideration.

4.3. Baselines

The existing methods used in these two datasets are different. In order to better compare the experiment in two datasets, we set up the most appropriate baselines for each of the datasets accordingly.

4.3.1. Baselines for SemEval-2010 Task 8

The baselines used in SemEval-2010 are recent published methods. They can be 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 et al. [4]. All of these methods use a considerable amount of resources (WordNet, ProBank, and FrameNet, for example) and employ SVM [22] or MaxEnt [23] as the classifier. The results of handcrafted feature based methods are shown in the first five rows of Table 4.

The neural network models are Convolutional Neural Network (CNN) based methods [7,9,13,25], Recursive Neural Networks(RecNN) based methods [5,27] and Long Short Term Memory Networks (LSTM) based methods [8,28].

- **CNN** [7] is the early work that exploits a convolutional deep neural network to extract lexical and sentence level features for the task of relation classification. Besides, it uses a special position vector that indicates the relative distances of current input word to two marked entities.
- **CR-CNN** [9] also applies convolutional neural network to classify relation classes. Instead of using softmax function on the top layer of CNN, [9] employs a pair-wise ranking strategy to reduce the effect of the confusing relation "Other".
- **depLCNN** [13] 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** [5] 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** [5] finds the path between the two entities in the constituent parse tree and learns the distributed representation of its highest node. It uses node's vector as feature to classify the relationship.

[2] http://www.itl.nist.gov/iad/mig//tests/ace/.

Please cite this article as: S. Zheng et al., A neural network framework for relation extraction: Learning entity semantic and relation pattern, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.09.019

Table 2
Hyper parameters of MixCNN+CNN.

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

Table 3
Hyper parameters of MixCNN+LSTM.

Parameter	Parameter description	Parameter value
d	Dimension of word embedding	300
$\|fh\|$	Hidden units number of forward layer	200
$\|bh\|$	Hidden units number of backward layer	200
ne	The filter number of CNN on ESE module	100
j	The number of CNNs on ESE module	4
ρ	The ratio of dropout in merged layer	0.3

- **SDP-LSTM** [8] 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 then input into a softmax classifier.

4.3.2. Baselines for ACE05

In recent years, neural network models have made significant progress in the task of relation classification. Therefore, the baselines used in ACE05 dataset are the representative neural networks.

- **CNN** [10] is the classic convolutional neural network with the sub-sentence as input.
- **BLSTM** [28] is a bidirectional LSTM networks to learn the sentence level features, and classify relations.
- **Att-BLSTM** [37] is the newest method to do the task of relation classifications. It is an attention-based bidirectional Long Short-Term Memory Networks, which can capture the most important semantic information in a sentence.

4.4. Hyper parameter settings

In this paper, we propose two kinds of neural network based model to extract relation. The hyper parameters used in MixCNN+CNN are summarized in Table 2 and MixCNN+LSTM's parameter are shown in Table 3. These two models share the same architecture of ESE module. Most parameters of ESE module in MixCNN+CNN and MixCNN+LSTM are kept same. Especially, in 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 is less than 5, the context window size is set to the length of the input.

5. Results

Most of works on relation classification only use the SemEval-2010 Task 8 dataset and few works use other dataset such as ACE05 dataset. Therefore, we set up the primary contrast experiment on SemEval-2010 Task 8 and use the contrast experiment on ACE05 to further illustrate the effectiveness of our method.

Table 4
Comparison of methods with adding different external resources on SemEval-2010 Task 8 dataset. 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[4]	POS, stemming, syntactic patterns	60.1
SVM[4]	word pair, words in between	72.5
SVM[4]	POS, stemming, syntactic patterns, WordNet	74.8
MaxEnt[4]	WordNet,Google n-grams,morphological...	77.6
SVM[4]	WordNet,FrameNet, morphological ...	82.2
RNN[5]	–	74.8
RNN[5]	POS, NER, WordNet	77.6
MVRNN[5]	–	79.1
MVRNN[5]	POS, NER, WordNet	82.4
FCM[6]	–	80.6
FCM[6]	Dependency parse, NER	83.0
SDP-LSTM[8]	–	82.4
SDP-LSTM[8]	POS, WordNet, Grammar relation	83.7
CNN[7]	–	69.7
CNN[7]	WordNet	82.7
depLCNN[25]	–	81.3
depLCNN[25]	Negative sampling	84.0
depLCNN[25]	WordNet	83.7
depLCNN[25]	WordNet,Negative sampling	**85.6**
CNN+softmax[9]	–	82.5
CNN+CR[9]	–	84.1
MixCNN+LSTM	–	83.8
MixCNN+CNN	–	**84.8**

5.1. Primary contrast experiment on SemEval-2010 Task 8

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. Different external resources have different effect on improving the predicted results. What's more, 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 4 shows.

In Table 4, only using word embedding as input features, MixCNN+CNN achieves F1 of 84.8%, which is the best results comparing with other methods. MixCNN+LSTM can also achieve F1 of 83.8%, when compared with most of baselines without adding different external resources. It shows that joint learning of entities' semantic properties and relation keywords is good for the task of relation classification. Besides, MixCNN+CNN is better than MixCNN+LSTM, which means that extracting the keyword information in the sub-sentence is more effective than representing sub-sentence's syntactical information.

Zeng et al. [7] is the early work that uses CNN to classify relation. Although Zeng et al. [7] can extract sentence level features, it cannot achieve good results when only using word embedding features. dos Santos et al. [9] 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 MixCNN+CNN and one percentage points worse than MixCNN+LSTM. Although MixCNN+CNN uses the softmax, it can be also superior to CR-CNN with the pairwise ranking loss function. Xu et al. [25] combines the dependency path and CNN to represent the sentence and can achieve the results of 81.3%.

Please cite this article as: S. Zheng et al., A neural network framework for relation extraction: Learning entity semantic and relation pattern, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.09.019

Table 5
Comparison results on ACE05 dataset without using any external resources.

Methods	*Prec.(%)*	*Rec.(%)*	*F1(%)*
CNN [10]	54.3	45.6	49.2
BLSTM [28]	52.6	45.9	48.4
Att − BLSTM[37]	57.64	44.35	49.6
MixCNN+LSTM	65.5	41.6	50.9
MixCNN+CNN	**60.0**	**48.4**	**53.6**

Table 6
Comparison of ACC and NMI of K-means cluster algorithm based on different relation representations on SemEval-2010 Task 8.

Dataset	Train		Test	
Relation embedding	R_v	R_{ese}	R_v	R_{ese}
ACC(%)	27.0 ± 1.1	76.2 ± 5.1	24.0 ± 1.0	60.1 ± 2.9
NMI(%)	22.2 ± 0.8	84.7 ± 1.9	20.6 ± 0.7	61.7 ± 0.9

Apart from the CNN methods, there are many sequential neural networks [5,8], which achieve results from 74.8% to 82.4%. Xu et al. [8] also uses the LSTM model to extract relationship. Instead of modeling the sub-sentence between the two entities, it leverages the shortest dependency path between two entities. MixCNN+LSTM can have 1.4% improvement when compared with SDP-LSTM [8] without using any resources. What's more, MixCNN+LSTM is also better than SDP-LSTM even if SDP-LSTM uses the resources: POS, WordNet, Grammar relation. It shows the importance of considering entities' semantic information.

We also compare our methods with this baselines by adding external resources in Table 4. 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 Xu et al. [25] only uses a negative sampling strategy to increase the number of training samples, MixCNN+CNN can still has *+0.8%* improvement over depLCNN. Besides, our method is better than depLCNN [25] when depLCNN only adds the WordNet information. If Xu et al. [25] uses WordNet and negative sampling strategy simultaneously on depLCNN, we can still get comparable results to theirs under the circumstance that our training set is the half of theirs and without using WordNet.

5.2. Auxiliary contrast experiment on ACE05

The results of different methods on ACE05 dataset are shown in Table 5. These methods all do not use any external information except word embedding. The texts in ACE05 are more complex than SemEval-2010, so the *F*1 results on ACE05 are worse than SemEval-2010. In spite of the difficulty of the task on ACE05, our methods can also achieve the state-of-the-art results when comparing with these representative neural networks. It shows the robustness and general applicability of our approaches. Our method $MixCNN + LSTM$ has a +1% improvement over other LSTM-based methods ($BLSTM$ [28], $Att - BLSTM$ [37]) and $MixCNN + CNN$ has a +4% improvement over the classic CNN [10] method. The main difference is that our model contains an entity semantic extraction module which can better capture the given entities information and improve the relation classification result.

5.3. 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 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 [20]. 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 produced 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 [38] 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) [39] and the normalized mutual information metric (NMI) [40], are used to measure the clustering performance [41]. 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:

$$ACC = \frac{\sum_{i=1}^{n} \delta(y_i, c_i)}{n}, \tag{13}$$

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:

$$NMI(\mathbf{Y},\mathbf{C}) = \frac{MI(\mathbf{Y},\mathbf{C})}{\sqrt{H(\mathbf{Y})H(\mathbf{C})}}, \tag{14}$$

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 [41].

We run 100 times for each experiment and obtain the final results on dataset SemEval-2010 Task 8 as Table 6 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} is significantly better than R_v.

We also visualize the clustering results by using t-SNE [42] as Fig. 6 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.

6. Analysis and discussion

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

Please cite this article as: S. Zheng et al., A neural network framework for relation extraction: Learning entity semantic and relation pattern, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.09.019

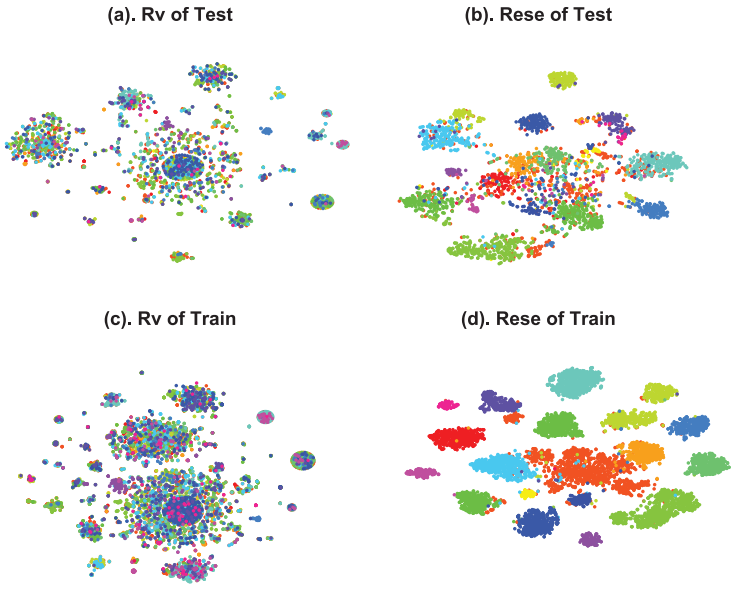

Fig. 6. The t-SNE visualization of the relation embeddings on SemEval-2010 Task 8. Figure (a) and Figure (c) are the relation embeddings produced by word2vec on training set and testing set. Figure (b) and Figure (d) are produced by ESE module.

Table 7
Comparisons of the RPE module with different configurations on SemEval-2010 Task 8.

Methods	*Prec.(%)*	*Rec.(%)*	*F1(%)*
$RPE1_{cnn}$	72.4	74.5	73.3
RPE_{cnn}	80.9	84.6	82.6
$RPE1_{lstm}$	69.90	72.14	71.4
RPE_{lstm}	81.6	85.0	83.2
$RPE1_{lstm} + ESE$	78.9	87.2	82.8
MixCNN+LSTM	79.1	89.4	83.8
$RPE1_{cnn} + ESE$	81.3	84.2	82.7
MixCNN+CNN	**83.1**	**86.6**	**84.8**

our method improves the final results on relation classification. Especially, SemEval-2010 Task 8 is the most widely used and authoritative dataset for relation classification task. Hence, the following analysis experiments are all conducted on the dataset.

6.1. RPE analysis

In the relation pattern extraction module (RPE), we set up two kinds of models to model the sub-sentence between the given two entities. The CNN-based model which focuses on extracting keyword information and the LSTM-based model which focuses on representing sentence's syntactical information.

Based on dos Santos et al. [9] and Xu et al.'s [8] analysis and our observations, extracting the sub-sentence information between given entities can better capture the relation pattern in the sentence when compared with learning a semantic embedding of the whole sentence. Hence, we also compare the results of full sentence configuration which is marked as RPE1. At first, we allow RPE module with different configurations to perform the task of relation classification. In addition to testing the effects of RPE module alone, we also test their combinations with ESE as Table 7 shows. In this paper, our method is the combination of RPE and ESE.

From Table 7, we know that RPE_{cnn} and RPE_{lstm} can also achieve comparable results of *F1* when compared with most of the baselines, which shows the validity of these two specific *RPE* models. Besides, when compared with RPE1 that extracts the relation pattern from full sentence, RPE can achieve a +10% improvement. When combined with ESE module, MixCNN+LSTM can have one percentage improvement than $RPE1_{lstm} + ESE$ and MixCNN+CNN have about two percentage improvement than $RPE1_{cnn} + ESE$. These results match our observations and dos Santos et al. [9] and Xu et al.'s [8] analysis that most relation patterns can be reflected by the sub-sentence between the given two entities.

The combination of RPE and ESE can bring about 10 points of improvement when compared with using RPE1 alone and can also better than using RPE alone. It shows the complementarity of the two modules as well as the necessity of module integration. Perhaps surprisingly, RPE_{lstm} is better than RPE_{cnn}, but MixCNN+CNN better than MixCNN+LSTM. One explanation is that the complementarity between RPE_{cnn} and ESE is better than that of RPE_{lstm} and ESE.

6.2. ESE analysis

We propose ESE module to extract the semantic properties of given entities based on their contextual words. In order to better illustrate the effectiveness of ESE module we proposed, we compare the ESE module with its variations. We directly use entity word embedding to represent entity information which denoted as EE. The ESE module uses a kind of mixture CNN to capture the semantic properties of entities. Hence, we also use a single CNN model to extract the semantic properties of entities based on different contextual words. The *CNNi* extracts entities' information based on the context that contains $2*i$ surrounding words of entity words. At first, we allow each variation to perform the task of relation classification alone. Then, we test their combinations with RPE_{cnn} as Table 8 shows.

From Table 8, we can see that ESE(MixCNN) achieves F1 of 83.1%, which is the best result when compared with other variations. Besides, the results of CNNs are all better than EE method which only uses the word embeddings. It shows the validity of mixture CNN and limitations of only using word embedings information on the task of relation classification. When combined with

Please cite this article as: S. Zheng et al., A neural network framework for relation extraction: Learning entity semantic and relation pattern, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.09.019

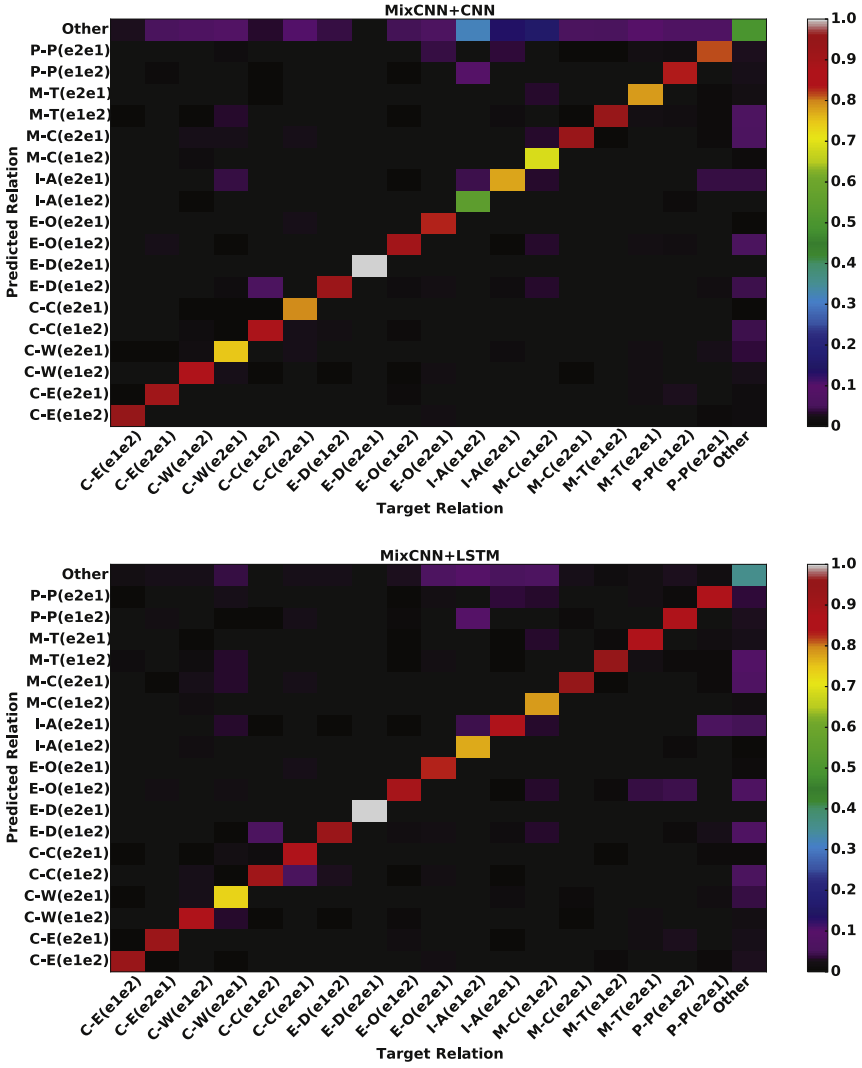

Fig. 7. The distribution of the predicted results for each relation class on SemEval-2010 Task 8. The horizontal axis is the target relation and each target relation corresponds to a column of predicted relations. Point (X,Y) means the ratio that the target relation is X and the predicted relation is Y. The sum of each column value equal to 1.

Table 8
Comparison of ESE module and its variations on SemEval-2010 Task 8.

Methods	*Prec.(%)*	*Rec.(%)*	*F1(%)*
EE	65.4	58.5	61.5
*CNN*1	72.1	73.9	72.9
*CNN*2	77.1	79.2	78.1
*CNN*3	80.5	83.4	81.9
*CNN*4	80.5	83.7	82.1
ESE(*MixCNN*)	81.8	84.6	83.1
$EE + RPE_{cnn}$	80.6	82.8	81.6
$CNN1 + RPE_{cnn}$	81.3	83.7	82.5
$CNN2 + RPE_{cnn}$	81.1	83.9	82.5
$CNN3 + RPE_{cnn}$	82.7	85.0	83.8
$CNN4 + RPE_{cnn}$	82.0	85.3	83.6
MixCNN+CNN	**83.1**	**86.6**	**84.8**

RPE_{cnn}, ESE and its variations are all improved. The combination of RPE_{cnn} and ESE achieves the best results. It further verifies the effectiveness of our framework.

6.3. 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 MixCNN+CNN's and MixCNN+LSTM's predicted results as Fig. 7 shows.

The diagonal region indicates the correct prediction results and the other regions reflect the distribution of error samples. From Fig. 7, we can see that these two specific models, which are based on our proposed framework, are consistent. The highlighted diagonal region means that our method can perform well on each relation class. However, 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. 7. 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)" performs 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 sen-

Please cite this article as: S. Zheng et al., A neural network framework for relation extraction: Learning entity semantic and relation pattern, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.09.019

tence. For examples: "Elevator(e1) operator(e2) is a meditation on the…" and "Camera(e1) operator(e2) is that person…". There is no indicative words between two entities and there are few contextual words around entities, So our framework is inadequate to deal with this case.

6.4. Discussion

The motivation of our work is to jointly learn the expression property of relations and entities' semantics, which is benefit for recognising entities' relationships. To achieve this target, we design a novel unified neural network framework for relation extraction and it contains RPE and ESE module.

RPE module aims to extract the expression property of relations, which are also called relation pattern here. The key words information and sentence's syntactical information both can indicate the relation pattern of a given sentence. Thus, we design two kinds of structures to extract key words information and syntactical information respectively. The CNN-based method contains a convolution layer, which can aggregate peripheral lexical information, and a max pooling layer that can extract the most significant features. Besides, we focus on the sub-sentence between two entities instead of the full sentence, which can narrow the range of extraction. Hence, CNN-based method can extract the key words information which imply the relation pattern. The other structure is LSTM-based method that is a recurrent neural network. The words in a sentence are input into the LSTM model in order. The LSTM-based model capture the syntactic properties of sentence and the long-distance relationships of words. Therefore, it can learn the syntactical characteristics of expression. When we compare these two structures with other configurations as Table 7, the results verify the effectiveness of our motivation.

Since the semantic properties of given entities are related to their contextual words, we set a novel mixture CNN model (MixCNN) to learn the semantics of entities. As we analyzed above, CNN is able to extract the key word information of a given sentence. Thus, each CNN in MixCNN focus on mining crucial background information from different ranges. Then a max pooling layer is used to merge these background information to get the most related features. Section 5.3 has proved that the ESE module can represent the semantic properties of the given entities effectively, when comparing word embeddings produced by word2vec. Besides, Table 8 also show that the mixed convolution manner is better than using a single CNN.

Furthermore, merging these two modules brings about 2 points of improvement in $F1$ value, which shows the complementarity of the two modules as well as the necessity of module integration.

7. Conclusion

In this paper, we propose a neural network based framework for the task of relation classification, which is an unified model by jointly learning entities' semantic properties and relation patterns. We set up two specific models based on this framework, the CNN-based model and LSTM-based model. The CNN-based model focuses on extracting keyword information that can reflect the relation pattern and the LSTM-based model focuses on representing sentence's syntactical information which can be beneficial to find the similar expressions. The models can achieve the state-of-the-art results on the SemEval-2010 Task8 dataset and the ACE05 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.

Acknowledgment

This work is also supported by the National High Technology Research and Development Program of China (863 Program) (Grant No. 2015AA015402), the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDB02070005) and National Natural Science Foundation (Grant No. 71402178 and Grant No. 61602479).

References

[1] N. Kambhatla, Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations (2004) 22–25.
[2] F.M. Suchanek, et al., Combining linguistic and statistical analysis to extract relations from web documents (2006) 712–717.
[3] I. Hendrickx, et al., Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals (2009) 94–99.
[4] B. Rink, et al., Utd: Classifying semantic relations by combining lexical and semantic resources (2010) 256–259.
[5] R. Socher, et al., Semantic compositionality through recursive matrix-vector spaces (2012) 1201–1211.
[6] M. Yu, et al., Factor-based compositional embedding models (2014).
[7] D. Zeng, et al., Relation classification via convolutional deep neural network (2014) 2335–2344.
[8] Y. Xu, et al., Classifying relations via long short term memory networks along shortest dependency paths (2015) 1785–1794.
[9] C.N. dos Santos, et al., Classifying relations by ranking with convolutional neural networks (2015) 626–634.
[10] Y. Kim, Convolutional neural networks for sentence classification (2014) 17461751.
[11] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural comput. 9 (8) (1997) 1735–1780.
[12] R. Collobert, et al., Natural language processing (almost) from scratch, J. Mach. Learn. Res. 12 (2011) 2493–2537.
[13] P. Blunsom, et al., A convolutional neural network for modelling sentences (2014) 655–665.
[14] S. Poria, et al., Aspect extraction for opinion mining with a deep convolutional neural network, Knowl. Based Syst. 108 (2016) 42–49.
[15] S. Ghosh, et al., Contextual lstm (clstm) models for large scale nlp tasks, arXiv preprint arXiv:1602.06291(2016).
[16] H. Palangi, et al., Deep sentence embedding using the long short term memory network: analysis and application to information retrieval (2015).
[17] J. Li, et al., A hierarchical neural autoencoder for paragraphs and documents (2015) 1106–1115.
[18] I. Sutskever, et al., Sequence to sequence learning with neural networks (2014) 3104–3112.
[19] M.T. Luong, et al., Addressing the rare word problem in neural machine translation (2015) 11–19.
[20] T. Mikolov, et al., Distributed representations of words and phrases and their compositionality (2013) 3111–3119.
[21] T.V.T. Nguyen, et al., Convolution kernels on constituent, dependency and sequential structures for relation extraction (2009) 1378–1387.
[22] M.A. Hearst, et al., Support vector machines, Intell. Syst. Appl. 13 (4) (1998) 18–28.
[23] S.J. Phillips, et al., Maximum entropy modeling of species geographic distributions, Ecol. Modell. 190 (3) (2006) 231–259.
[24] M. Pratama, J. Lu, et al., An incremental meta-cognitive-based scaffolding fuzzy neural network, Neurocomputing 171 (2016) 89–105.
[25] K. Xu, et al., Semantic relation classification via convolutional neural networks with simple negative sampling (2015) 536–540.
[26] D. Zhang, et al., Relation classification via recurrent neural network(2015).
[27] J. Ebrahimi, D. Dou, Chain based rnn for relation classification(2015) 1244–1249.
[28] S. Zhang, et al., Bidirectional long short-term memory networks for relation classification (2015) 73–78.
[29] L. Sun, X. Han, A feature-enriched tree kernel for relation extraction (2014) 61–67.
[30] Y. LeCun, et al., Gradient-based learning applied to document recognition, IEEE 86 (11) (1998) 2278–2324.
[31] X. Zhu, et al., Long short-term memory over recursive structures (2015) 1604–1612.
[32] A. Graves, Supervised Sequence Labelling, Springer, 2012.
[33] R. Kiros, et al., Skip-thought vectors (2015) 3276–3284.
[34] K. Duan, et al., Multi-category classification by soft-max combination of binary classifiers, in: The Journal of Multiple Classifier Systems, 2003, pp. 125–134.
[35] G.E. Dahl, et al., Improving deep neural networks for lvcsr using rectified linear units and dropout (2013) 8609–8613.
[36] L. Bottou, Stochastic gradient learning in neural networks, J. Neuro-Nımes 91 (1991).

[37] P. Zhou, W. Shi, et al., Attention-based bidirectional long short-term memory networks for relation classification (2016) 207–212.
[38] K. Wagstaff, et al., Constrained k-means clustering with background knowledge (2001) 577–584.
[39] D. Cai, et al., Document clustering using locality preserving indexing, IEEE Trans. Knowl. Data Eng. 17 (12) (2005) 1624–1637.
[40] W.Y. Chen, et al., Parallel spectral clustering in distributed systems, J. TPAMI 33 (3) (2011) 568–586.
[41] J. Xu, P. Wang, et al., Short text clustering via convolutional neural networks (2015) 62–69.
[42] L. Van der Maaten, G. Hinton, Visualizing data using t-SNE, J. Mach. Learn. Res. 9 (2008) 2579–2605. 85

Please cite this article as: S. Zheng et al., A neural network framework for relation extraction: Learning entity semantic and relation pattern, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.09.019