Information Processing and Management 54 (2018) 593–608

Contents lists available at ScienceDirect

Information Processing and Management

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

Using semantic similarity to reduce wrong labels in distant supervision for relation extraction

Chengsen Ru, Jintao Tang, Shasha Li, Songxian Xie, Ting Wang[*]

College of Computer, National University of Defense Technology, No.137 Yanwachi Street, Changsha, Hunan 410073, China

ARTICLE INFO

Keywords:
Distant supervision
Relation extraction
Wrong label
Semantic Jaccard
CNN

ABSTRACT

Distant supervision (DS) has the advantage of automatically generating large amounts of labelled training data and has been widely used for relation extraction. However, there are usually many wrong labels in the automatically labelled data in distant supervision (Riedel, Yao, & McCallum, 2010). This paper presents a novel method to reduce the wrong labels. The proposed method uses the semantic Jaccard with word embedding to measure the semantic similarity between the relation phrase in the knowledge base and the dependency phrases between two entities in a sentence to filter the wrong labels. In the process of reducing wrong labels, the semantic Jaccard algorithm selects a core dependency phrase to represent the candidate relation in a sentence, which can capture features for relation classification and avoid the negative impact from irrelevant term sequences that previous neural network models of relation extraction often suffer. In the process of relation classification, the core dependency phrases are also used as the input of a convolutional neural network (CNN) for relation classification. The experimental results show that compared with the methods using original DS data, the methods using filtered DS data performed much better in relation extraction. It indicates that the semantic similarity based method is effective in reducing wrong labels. The relation extraction performance of the CNN model using the core dependency phrases as input is the best of all, which indicates that using the core dependency phrases as input of CNN is enough to capture the features for relation classification and could avoid negative impact from irrelevant terms.

1. Introduction

Relation extraction task can be defined as follows: given a sentence S with a pair of entities $e1$ and $e2$, we aim to identify the relationship between $e1$ and $e2$ (Hendrickx et al., 2010). Relation extraction is a crucial step towards natural language understanding applications, i.e. question answering (Hazrina, Sharef, Ibrahim, Murad, & Noah, 2017) and knowledge graph (Franco-Salvador, Rosso, & y Gmez, 2016; Voskarides, Meij, Tsagkias, De Rijke, & Weerkamp, 2015). However, the methods of relation extraction often encounter problems with a lack of labelled data. It is time consuming to label training data manually. To save human effort, distant supervision (DS) is firstly used by Mintz, Bills, Snow, and Jurafsky (2009) for relation extraction, which can generate labelled training data automatically. It assumes that a sentence containing an entity pair in a knowledge base expresses the corresponding relation in the knowledge base. Under this assumption, a labelled training set can be automatically generated by checking the corpus to find all the sentences containing entity pairs of the known relations. This method has been a popular choice in relation extraction for saving human effort in labelling training data (Han & Sun, 2014; Hoffmann, Zhang, Ling, Zettlemoyer, & Weld, 2011; Min, Grishman, Wan,

[*] Corresponding author.
E-mail address: tingwang@nudt.edu.cn (T. Wang).

https://doi.org/10.1016/j.ipm.2018.04.002
Received 12 April 2017; Received in revised form 13 March 2018; Accepted 1 April 2018
Available online 22 April 2018

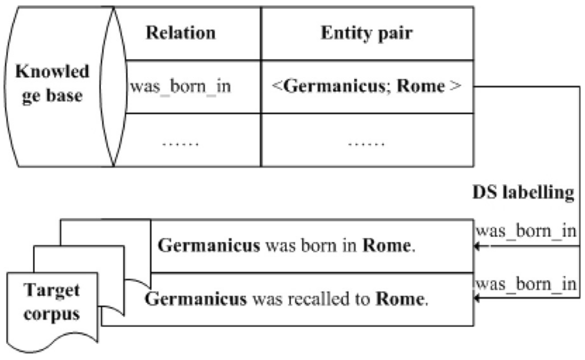


Fig. 1. The process of distant supervision.

Wang, & Gondek, 2013; Mintz et al., 2009; Riedel, Yao, & McCallum, 2010; Ritter, Zettlemoyer, Etzioni et al., 2013; Takamatsu, Sato, & Nakagawa, 2012).

However, the DS assumption may fail when there is more than one relation between an entity pair, which results in wrong labels. As shown in Fig. 1, both sentences contain the entity pair $< Germanicus, Rome >$, but express different relations, one expresses the relation *wasbornin*, the other expresses the relation *wasrecalledto*. In DS, both of them will be labelled with the same label of relation *wasbornin* or *wasrecalledto*. We have investigated the percentage of wrong labels introduced by DS in a real corpus which comes from a subset of Wikipedia containing 800,000 pages. The average error rate is 74.1% which may seriously affect the training of relation extraction models. If the wrong labels can be removed from the training data, it is expected to greatly improve the performance of relation extraction.

In the above example, we can see that the relation phrase of $< Germanicus, Rome >$ in knowledge base is *wasbornin*, and the dependency phrases on the dependency path between *Germanicas* and *Rome* in two sentences are *wasbornin* and *wasrecalledto* respectively. Obviously, the dependency phrases express the relation between *Germanicas* and *Rome*.

In most cases, the relation between two entities is described by the dependency phrases between them (Vo & Bagheri, 2017). Knowledge bases use relation phrases to describe various relations. Thus, it is possible to judge whether a sentence is correctly labelled by measuring the semantic similarity between the relation phrases and the dependency phrases. The higher the semantic similarity is, the greater the probability of correct labelling tends to be. Under this assumption, a semantic similarity based method is proposed to reduce the wrong labels in DS. The semantic Jaccard algorithm is used to calculate the semantic similarity between text fragments and has been proved effective(Zhang, 2016). In this paper, the semantic Jaccard is proposed to calculate the semantic similarity with the assumption of treating the relation phrases as a text fragment and all dependency phrases belonging to the same sentence as another text fragment. When the entity pair has a semantic Jaccard value greater than a certain similarity threshold, the sentence containing the entity pair is taken as true positive.

Besides the quality of training data, the performance of traditional relation extraction methods also heavily depends on the quality of the designed features which relies on the human ingenuity and prior NLP knowledge. In recent years, some neural network models have been developed for relation extraction as an automatic feature learning method and have been proved effective (Xu, Feng, Huang, & Zhao, 2015; Zeng et al., 2014). However, these methods all suffer from irrelevant term sequences as they pay little attention to context selection and need to set the window size of context terms, which is very difficult. In the process of reducing wrong labels, the semantic Jaccard would select a core dependency phrase to represent the given relation between the entities in a sentence, which can capture the feature for relation classification without setting the window size of context words and filter irrelevant terms. Therefore, we introduce a method which uses the core dependency phrases as input to a convolution neural network (CNN) model for relation classification.

The major contributions of the work presented in this paper are as follows:

1. We propose a semantic similarity based method to reduce wrong labels appearing in DS. The method uses the semantic Jaccard to measure the semantic similarity between the relation phrase in the knowledge base and the dependency phrases between the two entities. In the process of measuring similarity, the semantic Jaccard selects the core dependency phrase to represent the given relation between the entities in a sentence.
2. We propose to take the core dependency phrases as the input of a convolutional neural network (CNN) for relation classification. The core dependency phrases can exactly capture syntactic dependency context without setting the window size of context terms and avoid negative impact from irrelevant terms.

The remainder of the paper is organized as follows. Section 2 presents the related work. Section 3 introduces the target corpus and the knowledge base. Section 4 defines the Semantic Jaccard. Section 5 gives the objective and framework of this paper. Section 6 introduces the method to use the semantic Jaccard to reduce wrong labels and describes the CNN model for relation classification. Section 7 evaluates the performance of the proposed method. Section 8 gives the conclusion.

2. Related work

DS was first used in relation extraction by Mintz et al. (2009) with the intuition that if a sentence contains an entity pair in a knowledge base, it actually expresses the corresponding relation in the knowledge base.

DS only uses the known information of the knowledge base to generate the labelled data. It might generate a significant number of wrong negative examples because of the incompleteness of the knowledge base. To avoid this problem, Min et al. (2013) proposed an algorithm that learns from only positive and unlabelled instances. To reduce wrong negative examples directly, Xu, Hoffmann, Zhao, and Grishman (2013) proposed to combine a passage retrieval model using coarse features into the relation extractor. To model missing data in distant supervision, Ritter et al. (2013) proposed a joint model of information extraction and missing data which relaxes the hard constraints used in previous work to generate heuristic labels and provides a natural way to incorporate side information through a missing data model, for instance modelling the intuition that text will often mention rare entities which are likely to be missing in the knowledge base.

As described in Section 1, the assumption of distant supervision might fail as different sentences containing the same entity pair might express different relations. To relax the assumption of distant supervision, Riedel et al. (2010) and Hoffmann et al. (2011) used Multiple Instance Learning to tolerate the wrong positive labels. Their works did not reduce the wrong positive labels directly and might fail when a labelled entity pair appeared only once in the corpus but expressed another relation rather than the target relation. To reduce wrong positive examples directly, Takamatsu et al. (2012) proposed a generative model to estimate the probabilities of a pattern showing different relations and Han and Sun (2014) proposed a local subspace based method which modeled the local subspace around an instance as a sparse linear combination of training instances to identify reliable instances from noisy instances. The performance of their work is limited by the quantity of training instances and might perform poorly without sufficient instances.

In order to reduce wrong labels in DS, this paper proposes a novel method based on semantic similarity. Compared with the previous research, this method is not affected by the size and quality of labelled data. This method uses semantic Jaccard to measure the semantic similarity between relation phrases and the dependency phrases in each sentence to check whether each label is wrong.

The performance of all the above methods depends on the quality of the manually designed features used for training the relation classifiers. While neural networks has been widely used in NLP recently, some neural network models have also been developed for relation extraction. Socher, Huval, Manning, and Ng (2012) and Hashimoto, Miwa, Tsuruoka, and Chikayama (2013) used a recursive neural network (RNN) for relation classification. Zeng et al. (2014) and Xu et al. (2015) used a convolutional neural network (CNN) for relation classification. These methods took the representations of terms instead of the manually design features as input. Though these methods have been proved effective, they often suffered from irrelevant subsequences or clauses. For example, in the sentence,"The bomb, which ......, was discovered placed inside a car.", the which clause is used to modify *bomb*, but is irrelevant to the relation *placedinside* between *bomb* and *car*. In Zeng et al. (2014), such information will be incorporated into the extraction model and hurt the extraction performance. In this paper, we propose a method which uses the core dependency phrases in the *expandPath* (Wu & Weld, 2010) between subjects and objects as input to a CNN model and avoids negative impact from irrelevant information.

3. Target corpus and knowledge base

From Fig. 1, we can see that both target corpus and knowledge base play an important role in DS.

3.1. Wikipedia and YAGO

Wikipedia, a multi-language encyclopedia, has become the largest and most widely used electronic encyclopedia containing millions of documents and has an unparalleled advantage over other corpora in terms of quality and quantity. As a free encyclopedia that covers a wide range of areas, grows and updates very fast, Wikipedia provides a rich, reliable and low-cost content resource for extracting semantic relations and building a knowledge base. Based on this, the English Wikipedia is used as the target corpus in this paper.

YAGO is an on-line open knowledge base and mainly derived from Wikipedia. YAGO specifies the relation types and entity types. In YAGO, a relation instance is a fact triple consisting of two entities and a relation type. Currently, YAGO contains more than 10 million entities and more than 120 million facts about these entities.

Since almost all of the information in YAGO is derived from Wikipedia, relation instances in YAGO are more likely to be found in Wikipedia than in other knowledge bases, making it easier to obtain the annotation data. In order to get enough training data, we use YAGO2s as the source knowledge base. This paper selects four relations from YAGO2s to investigate the impact of wrong labels. The 4

Table 1
The details of the selected relations.

Relation	Semantics	The number of instances
wasbornin	the birthplace of somebody	218,757
diedin	the death place of somebody	54,174
isaffiliatedto	officially attach or connect (a subsidiary group or a person) to an organization	497,263
created	bring (something) into existence	278,455

Table 2
Details of the original positive examples.

Relation	size	Error rate
wasbornin	37,017	81.7%
diedin	22,405	95.6%
isaffiliatedto	29,817	50.2%
created	25,633	68.8%

relations are: *wasbornin*, *diedin*, *isaffiliatedto* and *created*. Table 1 shows the details of the four relations.

Using the DS, we finally generated a dataset called Wiki which contains 100,000 original positive examples, as shown in Table 1. The error rate in Table 2 is obtained by randomly sampling of the generated positive examples in each relation. We take 100 samples from each relation at a time, manually check whether each sample is correct and calculate the error rate for each relation. The final error rate in each relation is obtained by averaging the error rate of 10 sampling times. The average error rate of the four relations is 74.1% which is obviously high and will seriously affect the training of relation extraction models. If the wrong labels can be removed from the training data, it is expected to improve the performance of relation extraction. This investigation gives a strong motivation of study to reduce wrong labels.

3.2. New York Times and Freebase

The NYT dataset is a widely used dataset for DS (Riedel et al., 2010). In NYT, the target corpus is the New York Times and the knowledge base is Freebase (Bollacker, Evans, Paritosh, Sturge, & Taylor, 2008). Freebase is an online knowledge base that stores entities and their relations. In the dataset, four categories of relations ("people", "business", "person" and "location") are extracted from a December 2009 snapshot of Freebase, which contains over 1.8 million entities and over 3.2 million relation instances of 430 Freebase relation types. As Freebase entities are frequently mentioned in the New York Times, the New York Times is chosen as the target corpus.

The NYT dataset contains two sub datasets: Held-out and Manual datasets which both consist of a training set and a test set. In Held-out dataset, the training set contains 34,811 positive examples and 91,373 negative examples, the test set contains 6,444 positive examples and 166,004 negative examples. In Manual dataset, the training set contains 121,867 positive examples and 322,249 negative examples, the test set contains 502,202 unlabelled examples. We also have investigated the error rate of NYT dataset by using the same method used in Wiki dataset. The error rate of NYT is about 31.0%.

4. Semantic Jaccard

Jaccard is a commonly used algorithm for calculating text semantic similarity (Zhang, 2016). In original Jaccard, sentences are represented by the sets of n-grams. The Jaccard value is the ratio of the number of same n-grams in two sentences to the number of all n-grams in two sentences. The original Jaccard only uses literal matching that limits its use because of the lack of enough semantic information. To solve this problem, (Zhang, 2016) proposes the semantic Jaccard algorithm that uses word embedding to represent the semantics of terms in sentences. Formula (1) gives the definition of the semantic Jaccard.

$$\begin{cases} SemJac(X, Y) = \dfrac{s}{s+d}, (1.1) \\ s = \sum_{m} cosine(x_{sim}, y_{sim}), (1.2) \\ d = \sum_{n} (1 - cosine(x_{dif}, y_{dif})), (1.3) \\ cosine(x_{sim}, y_{sim}) \geq \delta, (1.4) \\ cosine(x_{dif}, y_{dif}) < \delta, (1.5) \\ m + n = min(|X|, |Y|), (1.6) \end{cases} \tag{1}$$

x_{sim} and x_{dif} represent n-grams of sentence X, y_{sim} and y_{dif} represent n-grams of sentence Y. s represents the semantic similar partition between sentences X and Y. Each n-gram appears only once. If a n-gram appears in formula (1.2), it will not appear in formula (1.3). δ in formula (1.4) and (1.5) indicates the similar threshold. m is the quantity of n-gram pairs whose cosine similarity is not smaller than the similar threshold δ. d represents the semantic difference partition between sentences X and Y. n is the quantity of n-gram pairs whose cosine similarity is smaller than the similar threshold δ. The sum of m and n equals to the minimum value of the quantity of n-grams in sentence X and Y.

Algorithm 1 shows the process of calculating semantic Jaccard. Given two sentences as follows:

$X = x1, x2, x3;$

$Y = y1, y2, y3;$

Input:

Semantic similarity matrix A(each element a_{ij} in matrix A represents the cosine similarity between the ith n-gram from sentence X and the jth n-gram from sentence Y);

similar threshold δ;

Output:

Semantic Jaccard value $SemJac$;

Loop the following steps until the algorithm exits:

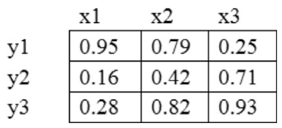
Step 1: Find the maximum value a_{ij} of all the current elements in the similarity matrix A.

Step 2: If a_{ij} is not smaller than the threshold δ, it is added to similarity part s; if a_{ij} is smaller than the δ, $1 - a_{ij}$ is added to the difference part d.

Step 3: Set all the elements in the ith row and the jth column to $-\infty$, which means that the two fragments are no longer involved in the subsequent calculation.

Step 4: If there is no element left, use formula (1.1) to calculate semantic Jaccard value $SemJac$ and exit; else, return to Step 1 to continue the process.

Algorithm 1. The process of calculating semantic Jaccard.



	x1	x2	x3
y1	0.95	0.79	0.25
y2	0.16	0.42	0.71
y3	0.28	0.82	0.93

Fig. 2. The semantic similarity matrix A between X and Y.

Fig. 2 shows the semantic similarity matrix A between X and Y.

When the similar threshold δ is set to 0.6, Fig. 3 shows the process of calculating semantic Jaccard between X and Y.

5. Objective and framework of our research

5.1. Objective

The objective of this study is to develop a semantic similarity based method for reducing the wrong labels in DS and for filtering irrelevant terms in the input of CNN model for relation extraction. It is possible to judge whether a sentence is correctly labelled by measuring the semantic similarity between the relation phrases and the dependency phrases. The semantic Jaccard holds the ability to measure the semantic similarity between the relation phrases in the knowledge base and the dependency phrases between the two entities. In order to reduce the wrong labels, we apply semantic Jaccard to find the instances more similar to the relation phrase and remove the ones with less similarity. In the process of reducing wrong labels, the semantic Jaccard would select a core dependency phrase to represent the given relation between the entities in a sentence, which can capture the feature for relation classification and filter irrelevant terms.

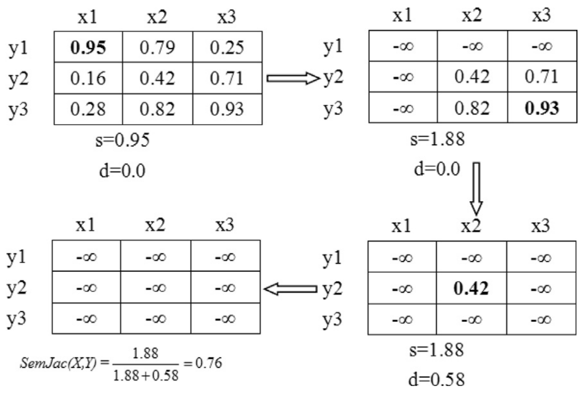
5.2. Framework

Fig. 4 shows the framework of our work, taking steps as follows:

1. Given the target corpus and knowledge base, we use DS to label training data and get the original labelled data.
2. Creating dependencies for sentences and getting dependency phrases in the *expandPath* between entities.
3. Using semantic Jaccard to measure the semantic similarity between the relation phrase in the knowledge base and the dependency phrases between the two entities and to get the core dependency phrase which represents the given relation between the entities in a sentence.
4. Compared with the similar threshold, the semantic similarity is used to reduce wrong labels and get the filtered data with high confidence core dependency phrases.
5. Using the core dependency phrases of the filtered data as the input of CNN for relation classification.

5.3. The expandPath and core dependency phrase

As mentioned in Wu and Weld (2010), there is a shortest dependency path called *corePath* from the first entity to the second that represents the relation of the entity pair. This paper uses the Dijkstra algorithm (Dijkstra, 1959) to get the *corePath* between the entity pair. While we would see that the terms in the *corePath* are useful for indicating when a relation exists between entities, they do not necessarily capture the semantics of that relation. In order to capture the meaning of the relation, we use terms in the *expandPath* which is a tree extended from the *corePath* by adding all the adverbial and adjectival modifiers to the *corePath*. The process of generating *expandPath* from *corePath* is shown in Algorithm 2.



	x1	x2	x3
y1	**0.95**	0.79	0.25
y2	0.16	0.42	0.71
y3	0.28	0.82	0.93

s=0.95
d=0.0

	x1	x2	x3
y1	-∞	-∞	-∞
y2	-∞	0.42	0.71
y3	-∞	0.82	**0.93**

s=1.88
d=0.0

	x1	x2	x3
y1	-∞	-∞	-∞
y2	-∞	**0.42**	-∞
y3	-∞	-∞	-∞

s=1.88
d=0.58

	x1	x2	x3
y1	-∞	-∞	-∞
y2	-∞	-∞	-∞
y3	-∞	-∞	-∞

$$SemJac(X,Y) = \frac{1.88}{1.88+0.58} = 0.76$$

Fig. 3. The process of calculating semantic Jaccard.

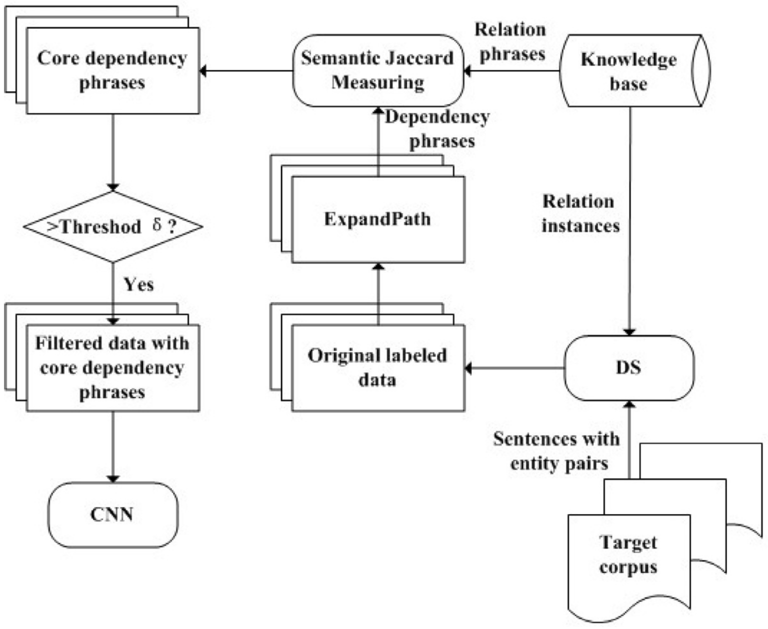


Fig. 4. The framework of our work.

Consider the sentence:

David was not born in Bethlehem.

Fig. 5., Fig. 6. respectively shows its *corePath* and *expandPath*.

In the above sentence, there is no relation $wasbornin$ between the entity pair *David* and *Bethlehem*. But term *born* is the only dependency term in the *corePath* from *David* to *Bethlehem*. Just depending on the term *born* could not capture the semantics that there is not a relation $wasbornin$. In order to catch the real meaning of the relation, we should also consider the modifiers *was, not* and *in* which modify the term *born* in the *expandPath*.

Definition 1. The dependency phrase: a word sequence consisting of a dependency term and its modifiers, e.g., the dependency term *born* and its modifiers *was, not* and *in* make up a dependency phrase $wasnotbornn$.

Definition 2. The core dependency phrase: the specific dependency phrase which represents the given relation between the entities in a sentence.

6. Methodology

6.1. The reduction of wrong labels

In this paper, we propose a method using semantic similarity to reduce the error ratio of the labelled data. The method assumes the dependency phrases in the *expandPath* between the entity pair provides semantic features of the relation between the entities. The key to pick out true positive examples is measuring the semantic similarity between the relation phrase and the dependency phrases between the two entities.

6.1.1. Representing term semantic by word embedding

In recent years, word embedding has been proved to be effective in representing the semantics of terms and can be applied to calculate the semantic similarity between words. Word embedding is often used in a vector form, where the value of each dimension corresponds to a feature and might even have a semantic or grammatical interpretation (Turian, Ratinov, & Bengio, 2010). In word embedding, words are embedded into a hyperspace, where two words that are more semantically similar to each other are located closer. This characteristic is precisely what this paper needs, because the key to reduce wrong labels is measuring the semantic similarity between the relation phrase and the dependency phrases between two entities. Based on this idea, this paper uses word embedding to represent the semantics of dependency phrases and relation phrases.

There have been many trained word embeddings freely available. This paper directly uses the trained embeddings provided by Turian et al. (2010).

A dependency phrase comprises of a dependency term and its all modifiers. The semantic representation of dependency phrases can be calculated by formula (2).

```
Input:
    expandPath←corePath;//the initiation of expandPath is corePath
    subIndex←the index of the subject;
    objIndex←the index of the object;
    n←the number of terms in a sentence;
    Term[i]←the ith term in a sentence;
    Dep[i][j]←the dependency relation between the ith term and the jth term;
    Path[i]←the index of term that the ith term directly points to in the corePath;
    Tag[i][j]←the tag for whether the dependency relation between the ith term and the jth term, 1 for yes, 0 for not;
Output:
    expandPath;
1: for each $i \in [subIndex, objIndex]$ do
2:   node←path[i]; //node is the index of the current term in the corePath;
3:   for each $j \in [1, n]$ do
4:     if Tag[node][j]==1 and j between i and path[node] then
5:       Add Term[j] and Dep[node][j] to expandPath; /* Term[j] is a child node of Term[node] and Dep[node][j] is the weight
         of the directed edge from Term[node] to Term[j]*/
6:     end if
7:   end for
8: end for
9: return expandPath;
```

Algorithm 2. The generation of expandPath.

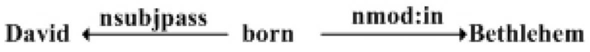


Fig. 5. An example of corePath.

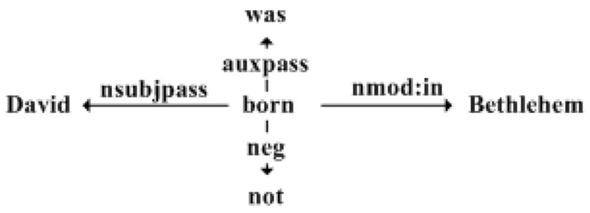


Fig. 6. An example of expandPath.

$$U_i = \alpha V_i + \beta \sum_{j=1}^{m} M_{ij} \tag{2}$$

In formula (2), U_i, $V_i \in R^n$ respectively represent the embedding vector for the ith dependency phrase and its unique dependency term t_i, $M_{ij} \in R^n$ represents the jth modifier that modifies the term t_i, α, β respectively represents the weight of term t_i and its modifiers, m is the number of modifiers, n is the dimension of the embedding vector. To distinguish the different effects of the dependency terms and modifiers in expressing the relation between entity pairs, α and β are set to 2 and 1 respectively.

For every relation, its relation phrase contains only one notional term and several modifiers. We can also use formula (2) to calculate the embedding of the relation phrase.

6.1.2. *Semantic similarity measuring by semantic Jaccard*

Unlike other measures of similarity making use of all parts for similarity measure without distinction, the semantic Jaccard treats the similar parts and the difference parts differently and can accurately obtain the similar parts. Using semantic Jaccard to measure the semantic similarity can get a better result. Thus, this paper uses the semantic Jaccard to measure the semantic similarity between the relation phrases and the dependency phrases of sentences. As a relation phrase or a dependency phrase represents a basic relation semantic unit, a relation phrase or a dependency phrase corresponds to an n-gram in formula (1). As the number of n-grams in a relation phrase is always 1, so we can use formula (3) instead of (1).

$$\begin{cases} SemJac(X, Y) = \begin{cases} 1, \ if \ \exists \ cosine(x_i, y_j) \geq \delta \\ 0, \ if \ \forall \ cosine(x_i, y_j) < \delta \end{cases} \\ -1 \leq \delta \leq 1 \end{cases} \tag{3}$$

Given a similar threshold δ, if there is a *cosine*(x_i, y_j) no smaller than the similar threshold δ, the corresponding sentence is marked with correct label; or not, the corresponding sentence is marked with wrong label.

6.2. *Relation classification via convolutional neural network model*

6.2.1. *Selection of core dependency phrase*

Given a relation, though an *expandPath* may contain more than a dependency phrase, only the core dependency phrase describes the specific relation between the subject and the object.

Consider the sentence:

The bomb was discovered placed inside a car.

Fig. 7 shows its *expandPath*. There are two dependency phrases *wasdiscovered* and *placedinside* in the above sentence. The entity pair *bomb* and *car* has the relation *wasplacedin*. In this case, using the shortest path or the expanded path between entities to describe their relation would bring in irrelevant information. The core dependency phrase *placedinside* is enough to capture features for relation classification.

In the process of using semantic Jaccard to measure semantic similarity, the core dependency phrase is selected to represent the

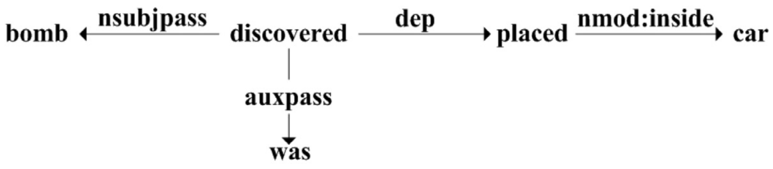


Fig. 7. An example of expandPath.

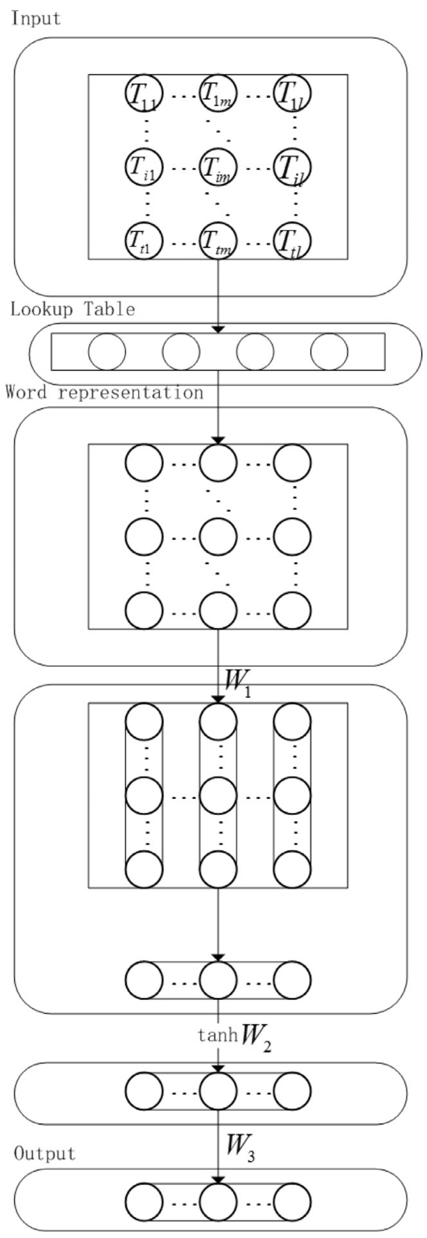


Fig. 8. The architecture of the CNN.

given relation in a sentence. The core dependency phrases are fed to a CNN model to capture features for relation classification and expected to filter out the negative impact of irrelevant terms.

6.2.2. *The architecture of CNN*

Fig. 8 describes the CNN architecture for relation classification. It mainly includes the following three parts: word representation, feature extraction and output. The network takes the core dependency phrases as input and discovers multiple levels of features, with higher levels representing more abstract aspects of the inputs.

In word representation part, each term is represented by a vector by looking up the embedding matrix $W_e \in R^{d \times |V|}$, where d is the dimension of a embedding vector and V is a set of all terms appearing in the corpus. Every phrase is represented by a embedding matrix of fixed size $d_p \times 1$, where $d_p = d \times 1$ and l is the maximum length of all core dependency phrases. This matrix can be built by concatenating the vectors of terms within a phrase.

In feature extraction part, the network uses two convolutional layers to extract features. The first convolutional layer uses a linear transformation $W_1 \in R^{n_1 \times d_p}$ to extract local feature from the embedding matrix, where n_1 is the number of the convolutional kernels in the first convolutional layer. The resulting matrix Z has size of $n_1 \times t$, where t is the number of the input dependency phrases.

$$Z = W_1 X + b_1 \tag{4}$$

The second convolutional layer uses hyperbolic *tanh* as the non-linearity function to extract more meaningful features. $W_2 \in R^{n_2 \times n_1}$ is the linear transformation matrix, where n_2 is the number of the convolutional kernels in the second convolutional layer. The output vector can be considered as higher level syntactic features, which is then fed to a softmax classifier used to compute the confidence of each relation. The output of the classifier is a vector, each dimension of which represents the confidence score of the corresponding relation.

$$Z_f = tanh(W_2 Z + b_2) \tag{5}$$

The softmax classifier is used to predict a K-class distribution $d(x)$, where K is the size of all possible relation types and the transformation matrix is $W_3 \in R^{K \times n_2}$. We denote $t(x) \in R^{K \times 1}$ as the target distribution vector: the entry $t_k(x)$ is the probability that the core dependency phrase describes the k-th relation.

To learn the network parameters, we optimize the cross-entropy error between $d_k(x)$ and $t_k(x)$ using stochastic gradient descent (SGD). Given all training instances, we define the objective as:

$$J(\theta) = -\sum_x \sum_{k=1}^{K} t_k(x) log(d_k(x)) \tag{6}$$

where θ represents the parameters need to learn. Gradients are computed using back propagation. We minimize the log likelihood $J(\theta)$. W_1, b_1, W_2, b_2 and W_3 are randomly initialized.

7. Results

7.1. Experiment settings

Held-out evaluation and manual evaluation are two kinds of evaluations of the performance of relation extraction in distant supervision (Mintz et al., 2009). Held-out evaluation sets aside some of the relation instances during training, and compares newly discovered relation instances against the held-out instances. Manual evaluation needs to manually check each predicted relation instance and mark whether the relation indeed holds between the entities.

Our experiments are conducted on two datasets: Wiki (constructed from Wikipedia and YAGO in this paper)and NYT (a widely used dataset that was developed in Riedel et al. (2010)).

7.1.1. Semantic similarity measuring

In order to verify the effectiveness of semantic Jaccard to calculate semantic similarity, we compare the following methods: semJac (semantic Jaccard); semCos (using cosine similarity with word embedding to measure semantic similarity). We randomly take 100 samples from all the generated positive examples of the dataset at a time and manually check whether each sample is correct. By changing the value of the similar threshold from 0 to 0.9, we got the precision. The final precision is obtained by averaging the precision of 10 sampling times. Fig. 9 and Fig. 10 respectively show the precision for different similar threshold of dataset Wiki and NYT. In both datasets, the performance of method semJac is obviously better than that of method semCos. In both pictures, The curve of method semCos only draws to 0.6 because there are no instances whose cosine similarity is bigger than 0.6.

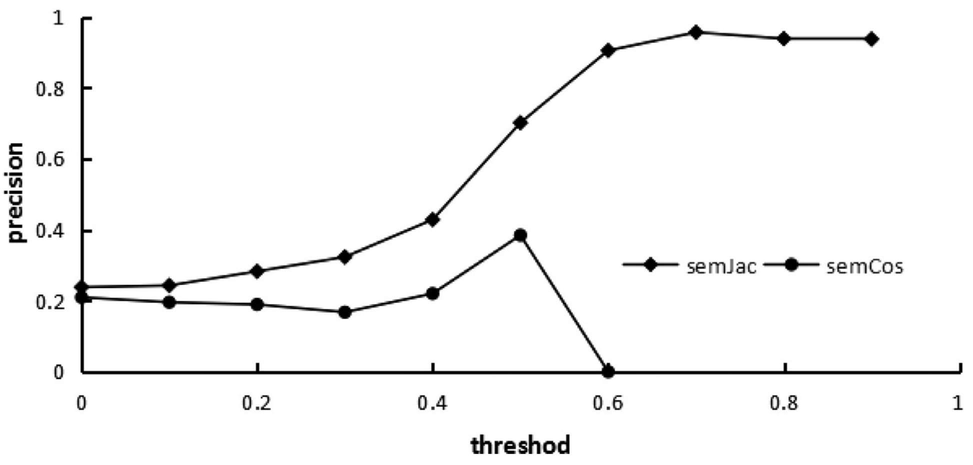


Fig. 9. The precision of positive examples by varying the similar threshold on Wiki dataset.

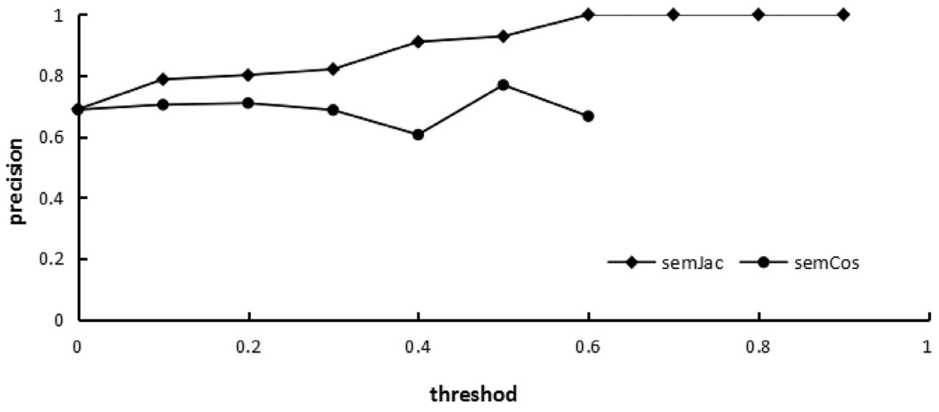


Fig. 10. The precision of positive examples by varying the similar threshold on NYT dataset.

7.2. Experiment 1 on Wiki

In our experiments, we randomly select 850,000 Wikipedia articles, each of which contains more than 5000 words: 800,000 for training, 50,000 for testing. In Wikipedia articles, named entities are identified by Stanford Named Entity Recognizer[1]. The Stanford Parser[2] is used to create dependencies for sentences in the *CCprocessedDependency* format.

We evaluated our methods in two ways: the held-out evaluation and the manual evaluation. In the manual evaluation, each predicted relation instance was labelled as true or false by 3 labellers. We assigned a value of true or false to each instance according to the majority vote of the labelers. We respectively compared the precision of the best 50, 100, 200 (according to their classification confidence) predicated relation instances of every relation in different methods.

In the Wikipedia articles used for testing, the sentences which contain at least two entities are selected for testing. We finally got a test set comprising of 1,100,000 sentences. We divided the sentences into two parts: 400,000 sentences for validation and the others for testing. The best similar threshold setting was obtained with values: 0.7, 0.7, 0.4 and 0.4, respectively for relation *wasbornin*, *diedin*, *isaffiliatedto* and *created*. In the experiment, we used the best similar thresholds to filter the positive training examples of the four relations. We got the error rate of the filtered positive examples in the same way as the error rate in the original positive examples. The error rates for relation *wasbornin*, *diedin*, *isaffiliatedto* and *created* were 4.3%, 2.0%, 18.2% and 30.5% respectively. The average error rate for the four relations dropped from 74.1% to 13.8%.

7.2.1. Experiment 1.1: reducing the wrong labels

In this experiment, we compare the following methods: LR (Mintz et al., 2009) (using the original DS labelled data to train a logistic regression model optimized using L-BFGS with Gaussian regularization); MultiR[3] (Hoffmann et al., 2011) (using the original DS labelled data to train a multi-instance learning system); FLR(using the filtered DS labelled data to train a logistic regression model optimized using L-BFGS with Gaussian regularization); FMultiR(using the filtered DS labelled data to train a multi-instance learning system). All the methods use syntactic features and lexical features developed by Mintz et al. (2009) to train models.

Manual evaluation

As Table 3 shows, the performance of methods FMultiR and FLR is significantly better than that of other methods. The results showed that the average precision of top50, top100 and top200 of FLR were respectively improved by 175%, 279% and 411% compared with LR in the four relations while those of FMultiR were 348%, 171%, 107% compared with MultiR. **It indicates that the method of using semantic similarity to reduce wrong labels is effective.**

In general, the method FMultiR performs best and MultiR is better than LR especially in top100 and top200. The use of multi-instance learning can reduce the impact of mislabelling to a certain extent.

The performance of method MultiR is worse than that of FLR. The results showed that the average precision of top50, top100 and top200 of FLR were respectively improved by 394%, 230% and 166% compared with MultiR. **It indicates that although multi-instance learning can reduce the effect of mislabelling to a certain extent, it will still be affected by wrong labels when the error is more frequent.**

The performance of method LR is the worst. As shown in Table 3, method LR can hardly extract relation *diedin*. By randomly sampling and manually labelling, it is found that more than 95% of the original DS labelled data in relation *diedin* is mislabelled. A large amount of noise is introduced in training extraction model, which seriously affects the performance of relation extraction.

In the four relations, the performance in relation *diedin* is worst when using the original DS labelled data to train models. By comparison, it is found that the error rate of the original training labelled data of relation *diedin* is the highest. This further indicates

[1] http://nlp.stanford.edu/software/CRF-NER.html.

[2] http://nlp.stanford.edu/software/lex-parser.html.

[3] For MultiR, we use the authorss implementation from http://www.cs.washington.edu/homes/raphaelh/mr/.

Table 3
Manual evaluation on Wiki dataset, precision of relations respectively in best 50, 100, 200 relation instances is reported.

Relation	Method	50	100	200
wasbornin	FLR	0.68	0.8	0.845
	FMultiR	0.88	0.89	0.845
	FDCNN	0.84	0.90	0.895
	FPDCNN	**0.90**	**0.93**	**0.965**
	LR	0.44	0.29	0.21
	MultiR	0.18	0.38	0.435
	DCNN	0.74	0.87	0.915
	PDCNN	0.72	0.86	0.835
diedin	FLR	0.8	0.49	0.35
	FMultiR	0.9	0.76	0.58
	FDCNN	**1.0**	0.97	0.97
	FPDCNN	0.98	**0.99**	**0.975**
	LR	0	0.01	0.005
	MultiR	0.18	0.24	0.265
	DCNN	0.22	0.18	0.19
	PDCNN	0.86	0.91	0.85
isaffiliatedto	FLR	0.82	0.91	**0.955**
	FMultiR	0.64	0.8	0.895
	FDCNN	0.90	0.93	0.95
	FPDCNN	**0.94**	**0.97**	0.91
	LR	0.24	0.18	0.09
	MultiR	0.18	0.27	0.36
	DCNN	0.46	0.57	0.59
	PDCNN	0.5	0.55	0.645
created	FLR	0.12	0.15	0.125
	FMultiR	0.18	0.21	0.22
	FDCNN	0.20	**0.33**	0.255
	FPDCNN	**0.28**	0.30	**0.265**
	LR	0.2	0.14	0.095
	MultiR	0.04	0.09	0.17
	DCNN	0.06	0.10	0.08
	PDCNN	0.22	0.2	0.23

that the quality of training data directly relates to the performance of relation extraction. At the same time, after reducing wrong labels and using filtered data to train models, the performance in relation *diedin* is improved most obviously. **This shows that the method of reducing wrong labels based on semantic similarity is effective, which improves the performance of relation extraction.**

The performance of all methods in relation *created* is not good as in other relations. After checking the data, we find that the filtered positive examples contains a portion of examples which express relation *ownedby*. It indicates that depending on the word embeddings can differentiate most terms which have different meanings but not all. In this situation, we need external resources to distinguish them.

Held-out evaluation

In held-out evaluation shown in Fig. 11, the performance of method FLR is better than method LR and the performance of FMultiR

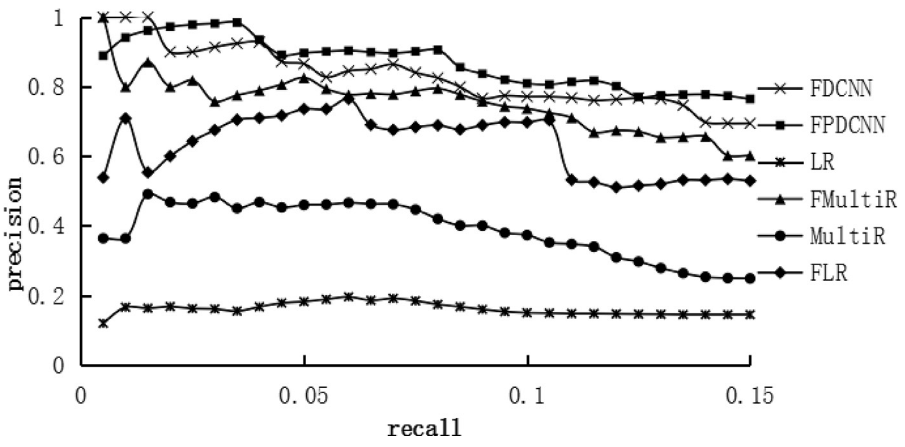


Fig. 11. The precision/recall curves on Wiki dataset.

is the best. **This shows that the method of reducing wrong labels based on semantic similarity is effective, which improves the performance of relation extraction.**

In held-out evaluation shown in Fig. 11, the performance of method FLR is better than method MultiR. **This shows that although multi-instance learning can reduce the impact of wrong labels to a certain extent, when the wrong labels is frequent, it will still be affected by wrong labels.**

7.2.2. *Experiment 1.2: relation extraction via CNN*

We initialized W_e with 50-dimensional word vectors trained by Turian et al. (2010). We tuned the hyper parameters using the validation set for each experimental setting. The hyper parameters include l, n_1, n_2. The best setting was obtained with the values: 3, 200, 100.

We compared LR, MultiR, FLR and FMultiR with the following methods: DCNN, FDCNN (taking terms on the corePath with bag of words context as the input of CNN, proposed by Xu et al. (2015)); PDCNN, FPDCNN (taking the core dependency phrase in the expandPath as input, proposed in this paper). The methods DCNN and PDCNN use the original DS labelled data to train models while the methods FDCNN and FPDCNN use the filtered DS labelled data to train models. Table 3 illustrates the measure results for these competing methods.

Manual evaluation

As Table 3 shows, the performance of methods FPDCNN and FDCNN is better than that of other methods and the performance of methods PDCNN and DCNN is better than that of methods LR and MultiR. The average precision of top50, top100 and top200 of FPDCNN were respectively improved by 12%, 15% and 20% compared with the best performance of FLR and FMultiR in the four relations while those of FDCNN were 6%, 9%, 18% compared with the best performance of FLR and FMultiR. The average precision of top50, top100 and top200 of PDCNN were respectively improved by 117%, 145% and 108% compared with the best performance of LR and MultiR in the four relations while those of DCNN were 40%, 67%, 44% compared with the best performance of LR and MultiR. **It indicates that using CNN can automatically learn the features which yield excellent results in relation extraction and can replace the traditionally designed features.**

As Table 3 shows, the performance of method FPDCNN and is better than other method FDCNN and the performance of method PDCNN is better than method DCNN. The average precision of top50, top100 and top200 of FPDCNN were respectively improved by 5%, 2% and 1% compared with FDCNN in the four relations while those of PDCNN were 55%, 47%, 44% compared with DCNN. **It indicates that using the core dependency phrases as input of CNN is enough to capture the features for relation classification and could avoid negative impact from irrelevant terms to most extent.**

Held-out evaluation

In held-out evaluation shown in Fig. 11, the performance of methods FPDCNN and FDCNN is better than that of other methods. **It indicates that using CNN can automatically learn the features which yield excellent results in relation extraction and can replace the traditionally designed features.**

The performance of method FPDCNN is the best of all. **It indicates that using the core dependency phrases as input of CNN is enough to capture the features for relation classification and could avoid negative impact from irrelevant terms to most extent.**

7.3. *Experiment 2 on NYT*

We evaluated our methods in two ways: the held-out evaluation and the manual evaluation. The held-out evaluation compares the extracted relation instances against Freebase relation data and reports the precision/recall curves of the experiments. In the manual evaluation, the experiments are tested on the manually annotated data in Riedel et al. (2010). The best similar threshold was set to 0.1. The error rate dropped from 31.0% to 22.7%.

7.3.1. *Experiment 2.1: reducing the wrong labels*

In this experiment, we also compared the following methods: LR, MultiR, FLR and FMultiR.

Manual evaluation

In manual evaluation shown in Table 4, the performance of method FLR is better than method LR and the performance of FMultiR is the best. The precision of FLR was improved by 4.8% compared with LR while that of FMultiR was 1.5% compared with MultiR. **This shows that the method of reducing wrong labels based on semantic similarity is effective, which improves the performance of relation extraction.**

Held-out evaluation

In held-out evaluation shown in Fig. 12, the performance of method FLR is better than method LR and the performance of FMultiR is the best. **This shows that the method of reducing wrong labels based on semantic similarity is effective, which improves**

Table 4
Manual evaluation on NYT dataset.

Method	LR	FLR	MultiR	FMultiR	FDCNN	FPDCNN
Precision	68.8%	73.6%	77.6%	79.1%	80.5%	**81.2%**

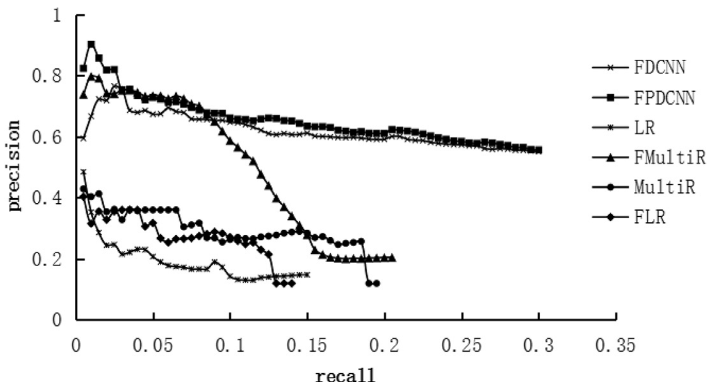


Fig. 12. The precision/recall curves on NYT dataset.

the performance of relation extraction.

7.3.2. *Experiment 2.2: relation extraction via CNN*

In this experiment, we compared the following methods: FLR, FMultiR, FDCNN and FPDCNN.

Manual evaluation

In manual evaluation shown in Table 4, the performance of methods FPDCNN and FDCNN is better than that of other methods. The precision of FPDCNN was improved by 7.6% and 2.1% respectively compared with FLR and FMultiR. The precision of FDCNN was improved by 6.9% and 1.4% respectively compared with FLR and FMultiR. **It indicates that using CNN can automatically learn the features which yield excellent results in relation extraction and can replace the traditionally designed features.**

The performance of method FPDCNN is better than that of method FDCNN. The precision of FPDCNN was improved by 0.7% compared with FDCNN. **It indicates that using the core dependency phrases as input of CNN is enough to capture the features for relation classification and could avoid negative impact from irrelevant terms to most extent.**

Held-out evaluation

In held-out evaluation shown in Fig. 12, the performance of methods FPDCNN and FDCNN is better than that of other methods. **It indicates that using CNN can automatically learn the features which yield excellent results in relation extraction and can replace the traditionally designed features.**

The performance of method FPDCNN is the best of all. **It indicates that using the core dependency phrases as input of CNN is enough to capture the features for relation classification and could avoid negative impact from irrelevant terms to most extent.**

8. Conclusions

This paper proposes a semantic similarity based method that reduces wrong labels appearing in DS. Our method uses semantic Jaccard with word embedding to measure the semantic similarity between relation phrases and dependency phrases among entity pairs in a sentence. Compared with the similar threshold, the semantic similarity is used to reduce wrong labels. Then, this paper uses the core dependency phrases output by semantic Jaccard as the input of CNN used for relation classification to avoid negative impact from irrelevant terms. The experimental results show that our method successfully reduced wrong labels and remarkably improved the performance of relation extraction.

Acknowledgment

The research is supported by the National Natural Science Foundation of China (61472436, 61532001).

References

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J. (2008). *Freebase: a collaboratively created graph database for structuring human knowledge. SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on management of data.* ACM1247–1250.
Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische mathematik, 1*(1), 269–271.
Franco-Salvador, M., Rosso, P., & y Gmez, M. M. (2016). A systematic study of knowledge graph analysis for cross-language plagiarism detection. *Information Processing and Management, 52*(4), 550–570 http://doi.org/10.1016/j.ipm.2015.12.004.
Han, X., & Sun, L. (2014). *Semantic consistency: A local subspace based method for distant supervised relation extraction. Proceedings of the 52th annual meeting of the association for computational linguistics*718–724.
Hashimoto, K., Miwa, M., Tsuruoka, Y., & Chikayama, T. (2013). *Simple customization of recursive neural networks for semantic relation classification. Proceedings of the empirical methods in natural language processing*1372–1376.
Hazrina, S., Sharef, N. M., Ibrahim, H., Murad, M. A. A., & Noah, S. A. M. (2017). Review on the advancements of disambiguation in semantic question answering system. *Information Processing and Management, 53*(1), 52–69 http://doi.org/10.1016/j.ipm.2016.06.006.
Hendrickx, I., Kim, S. N., Kozareva, Z., Nakov, P., Séaghdha, D. O., Padó, S., et al. (2010). *SemEval-2010 task 8: Multi-way classification of semantic relations between pairs*

*of nominals. Proceedings of the 5th international workshop on semantic evaluation*33–38.

Hoffmann, R., Zhang, C., Ling, X., Zettlemoyer, L., & Weld, D. S. (2011). *Knowledge-based weak supervision for information extraction of overlapping relations. Proceedings of the 49th annual meeting of the association for computational linguistics*541–550.

Min, B., Grishman, R., Wan, L., Wang, C., & Gondek, D. (2013). *Distant supervision for relation extraction with an incomplete knowledge base. Proceedings of the North American chapter of the association for computational linguistics*777–782.

Mintz, M., Bills, S., Snow, R., & Jurafsky, D. (2009). *Distant supervision for relation extraction without labeled data. Proceedings of the joint conference of the 47th annual meeting of the ACL and the 4th international joint conference on natural language processing of the AFNLP*1003–1011.

Riedel, S., Yao, L., & McCallum, A. (2010). *Modeling relations and their mentions without labeled text. Machine learning and knowledge discovery in databases*. Springer148–163.

Ritter, A., Zettlemoyer, L., Etzioni, O., et al. (2013). Modeling missing data in distant supervision for information extraction. *Transactions of the Association for Computational Linguistics, 1*, 367–378.

Socher, R., Huval, B., Manning, C. D., & Ng, A. Y. (2012). *Semantic compositionality through recursive matrix-vector spaces. Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*1201–1211.

Takamatsu, S., Sato, I., & Nakagawa, H. (2012). *Reducing wrong labels in distant supervision for relation extraction. Proceedings of the 50th annual meeting of the association for computational linguistics*721–729.

Turian, J., Ratinov, L., & Bengio, Y. (2010). *Word representations: a simple and general method for semi-supervised learning. Proceedings of the 48th annual meeting of the association for computational linguistics*384–394.

Vo, D.-T., & Bagheri, E. (2017). Self-training on refined clause patterns for relation extraction. *Information Processing and Management* http://doi.org/10.1016/j.ipm.2017.02.009.

Voskarides, N., Meij, E., Tsagkias, M., De Rijke, M., & Weerkamp, W. (2015). *Learning to explain entity relationships in knowledge graphs. Proceedings of the 53th annual meeting of the association for computational linguistics*564–574.

Wu, F., & Weld, D. S. (2010). *Open information extraction using wikipedia. Proceedings of the 48th annual meeting of the association for computational linguistics*118–127.

Xu, K., Feng, Y., Huang, S., & Zhao, D. (2015). *Semantic relation classification via convolutional neural networks with simple negative sampling. Proceedings of the empirical methods in natural language processing*536–540.

Xu, W., Hoffmann, R., Zhao, L., & Grishman, R. (2013). *Filling knowledge base gaps for distant supervision of relation extraction. Proceedings of the 51th annual meeting of the association for computational linguistics*665–670.

Zeng, D., Liu, K., Lai, S., Zhou, G., Zhao, J., et al. (2014). *Relation classification via convolutional deep neural network. Proceedings of the international conference on computational linguistics*2335–2344.

Zhang, J. (2016). One of the poor semantic processing toolbox: the semantic Jaccard. http://blog.csdn.net/malefactor/article/details/50471118.