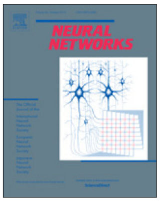
Neural Networks 108 (2018) 240–247

Contents lists available at ScienceDirect

Neural Networks

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



Distant supervision for relation extraction with hierarchical selective attention

Peng Zhou [a,b], Jiaming Xu [a], Zhenyu Qi [a,*], Hongyun Bao [a], Zhineng Chen [a], Bo Xu [a,b,c]

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

ARTICLE INFO

Article history:
Received 12 December 2017
Received in revised form 7 August 2018
Accepted 20 August 2018
Available online 29 August 2018

Keywords:
Relation extraction
Distant supervision
Hierarchical attention
Piecewise convolutional neural networks

ABSTRACT

Distant supervised relation extraction is an important task in the field of natural language processing. There are two main shortcomings for most state-of-the-art methods. One is that they take all sentences of an entity pair as input, which would result in a large computational cost. But in fact, few of most relevant sentences are enough to recognize the relation of an entity pair. To tackle these problems, we propose a novel hierarchical selective attention network for relation extraction under distant supervision. Our model first selects most relevant sentences by taking coarse sentence-level attention on all sentences of an entity pair and then employs word-level attention to construct sentence representations and fine sentence-level attention to aggregate these sentence representations. Experimental results on a widely used dataset demonstrate that our method performs significantly better than most of existing methods.

© 2018 Elsevier Ltd. All rights reserved.

1. Introduction

Distant supervised relation extraction aims to predict semantic relations between pairs of entities in texts supervised by Knowledge Bases (KB). It plays a significant role in various Natural Language Processing (NLP) tasks, such as question answering (Bian, Liu, Agichtein, & Zha, 2008; Sun et al., 2015) and knowledge base construction (Dong et al., 2014; Sa et al., 2016).

Normally, relation facts are formatted as triples in KB. For a triplet $r(e1, e2)$ in KB, all sentences containing entities $e1$ and $e2$ will be regarded as the training instances of relation r and these sentences constitute a bag with the relation r as the label. Table 1 shows the training instances of the triplet */business/commany/founders (Microsoft, Bill Gates)*.

As a traditional method, Mintz, Bills, Snow, and Dan (2009) assumed that if two entities have a relationship in a known knowledge base, then all sentences that mention these two entities would express that relationship in some way. For example, */business/commany/founders (Microsoft, Bill Gates)* is a triplet in Freebase, the sentences from S_1 to S_6 will be regarded as valid instances for relation */business/company/founder*. However, the last two sentences (S_5 and S_6) do not express the relation */business/company/founder*. The assumption is too strong and would cause the wrong labelling problem. Hoffmann, Zhang, Ling, Zettlemoyer, and Weld (2011) and Surdeanu, Tibshirani, Nallapati, and Manning (2012) relaxed this assumption and adopted multi-instance learning (Dietterich, Lathrop, & Lozanoperez, 1997) to alleviate the wrong labelling problem.

Previous methods (Hoffmann et al., 2011; Mintz et al., 2009; Surdeanu et al., 2012) typically applied supervised models to elaborately designed features when obtained the labeled data through distant supervision. The main weakness of these methods is that most features are explicitly derived from NLP tools such as dependency parser and the errors generated by NLP tools would propagate in these methods.

Motivated by the successful utilization of deep neural networks in speech recognition (Hinton et al., 2012), computer vision (Schmidhuber, Meier, & Ciresan, 2012) and NLP (Bengio, Schwenk, Sencal, Morin, & Gauvain, 2006), some neural distant supervised relation extraction models are proposed to learn low-dimensional text features without any feature engineering (Ji, Liu, He, & Zhao, 2017; Lin, Shen, Liu, Luan, & Sun, 2016; Zeng, Liu, Chen, & Zhao, 2015). Zeng et al. (2015) combined multi-instance learning and Piecewise Convolutional Neural Networks (PCNN) to choose the most likely valid sentence for each entity pair. Lin et al. (2016) proposed a sentence-level attention-based model to select the valid instances. Ji et al. (2017) regarded entity description derived from Freebase and Wikipedia pages as background knowledge to provide more information for predicting relations and bring better entity representations for the sentence-level attention-based model.

However, there are two major shortcomings for these methods (Ji et al., 2017; Lin et al., 2016; Zeng et al., 2015). First, the computational cost is very high because they take all instances in a bag as

[*] Corresponding author.
E-mail address: zhenyu.qi@ia.ac.cn (Z. Qi).

https://doi.org/10.1016/j.neunet.2018.08.016
0893-6080/© 2018 Elsevier Ltd. All rights reserved.

Table 1
Training instances of the triplet /business/company/founders (Microsoft, Bill Gates).

/business/company/founders (Microsoft, Bill Gates):
(S_1) [Bill Gates] and Paul Allen founded [Microsoft] on April 4, in 1975.
(S_2) [Bill Gates], the founder of [Microsoft], donated to his foundation more than $3 billion.
(S_3) [Bill Gates], the co-founder of [Microsoft], is now ranked as the world's first richest person.
(S_4) Last month, [Microsoft] 's co-founder, [Bill Gates], announced that he would leave his day-to-day role at the company in two years.
(S_5) [Bill Gates] has stepped down as chairman of [Microsoft] to take a more active role in the business.
(S_6) [Microsoft] was 25 years old before [Bill Gates] set up his foundation, which is a tax-exempt organization and separate from Microsoft.

input, and the bag maybe contains tens of thousands of sentences. For example, the bag in a benchmark distant supervision dataset developed by Riedel, Yao, and Mccallum (2010) contains up to 5000 sentences. However, dozens of relevant sentences is enough to recognize the relation of an entity pair. Second, they treat all words in a sentence as the same important and ignore the fact that the keywords are more crucial for the sentence meaning than other words in the sentence. For example, the keywords *"founded", "founde" and "co-founder"* have particular significance for the relation */business/company/founder* in the sentences S_1, S_2, S_3, S_4.

In order to address the above issues, we propose a Hierarchical Selective Attention Network (HSAN) for distant supervision relation extraction. Our HSAN first selects several relevant sentences by taking coarse sentence-level attention on each sentence in a bag with the given relation. Then, it utilizes PCNN, word-level attention to extract word-level features and construct sentence representations. Finally, it employs fine sentence-level attention to form the bag representation by aligning a different score to each sentence representation, which is fed into a softmax classifier to predict the relation. The experimental results on a real-world popular dataset show that our model outperforms most of the existing methods.

The contributions of this paper can be summarized as follows:

1. To reduce the computational cost, we propose a hierarchical selective attention network, where we only select most relevant sentences via coarse sentence-level attention and then conduct reason over them to predict the relation.
2. Considering the fact that the keywords are more important to the sentence meaning than other words in the sentence, we employ a word-level attention mechanism to select multiple valid words in the sentence.
3. We conduct experiments on a real-world popular dataset developed by Riedel et al. (2010), and the results indicate that our proposed model HSAN outperforms a range of baselines.

2. Related work

In relation extraction we often encounter a lack of explicitly annotated text, but an abundance of structured data source such as large scale public knowledge bases like Freebase. Distant supervision methods for relation extraction provide an effective solution to make full use of KB and unstructured text, they heuristically align the given knowledge base to text and use this alignment to learn a relation extractor. Since they do not rely on annotated text and KB grows fast recently, they have appealed much attention, and various types of models have been proposed.

Mintz et al. (2009) aggregated features from all instances in a bag and then fed them into a classifier, which caused the wrong label problem. Riedel et al. (2010) assumed that if two entities participate in a relation, at least one sentence that mentions these two entities might express that relation and utilized an undirected graphical model to predict which sentences express the relation. Based on the Multi-Instance Learning (Dietterich et al., 1997), Hoffmann et al. (2011) and Surdeanu et al. (2012) introduced a probabilistic, graphical model to select sentences and allowed for overlapping relations.

Recently, deep neural networks can learn underlying features automatically and have been used in the literature. Most representative progress was made by Zeng et al. (2015), who first incorporated multi-instance learning with PCNN. But they selected only the most likely sentence for each entity pair in training and testing without making full use of the information in the neglected sentences. Lin et al. (2016) employed PCNN to learn sentence representations and sentence-level attention to aggregate these representations and achieved state-of-the-art performance on the dataset developed by Riedel et al. (2010). Ji et al. (2017) utilized a sentence-level attention module to select the valid instances and entity descriptions from Freebase and Wikipedia pages as the background knowledge, which not only provides more information for predicting relations, but also brings better entity representations for the attention module. Zeng, Lin, Liu, and Sun (2017) proposed a path-based neural relation extraction model to encode the relational semantics from both direct sentences and inference chains between two target entities via intermediate entities. Jiang, Wang, Li, and Wang (2016) utilized cross-sentence max-pooling to select features across different sentences, and then aggregated the most significant features for each entity pair. Zeng, Zeng, and Dai (2017) exploited cost-sensitive ranking loss to alleviate the class imbalance problem.

The proposed model HSAN is relevant to PCNN+ATT (Lin et al., 2016). There are two differences between these two models. One is that PCNN+ATT takes all instances in a bag as input, while our model HSAN selects several instances related to the label of the bag and reasons on these selected instances to predict a relation. Another is that PCNN+ATT gives the same weight to the words in a sentence and ignore the fact that words are differentially important, while HSAN employs a word-level attention mechanism to dynamically highlight important parts of the sentence.

HSAN is also relevant to PCNN+PF (Qu, Ouyang, Hua, Ye, & Li, 2018), which also uses a word-level attention-based mechanism to determine the critical words to construct a more informative sentence representation. The main difference between the two models is that HSAN utilizes a coarse sentence-level attention mechanism to select several relevant sentences and predicts a relation to the bag based on the selected sentences, while PCNN+PF uses all sentences in a bag to predict the relation, so its computational cost is much higher than HSAN.

3. Methodology

As shown in Fig. 1(a), we give an illustration of our model HSAN. Given a set of l sentences denoted as: $B = \{S_1, S_2, \ldots, S_l\}$ and two corresponding entities: e_1 and e_2, we first map these sentences to sentence representations with low-dimensional distributed vectors. Then, we retrieve these sentence representations to soft-search the related sentences. We further exploit PCNN and word-level attention to automatically learn features based on the soft-searching results and concatenate these features as the new sentence representation. We then use fine sentence-level attention to select the sentences which really express the corresponding relation r and aggregate them as the bag representation. Finally, we feed the bag representation into a softmax classifier to predict the relation. We will describe details and the learning objective of HSAN below.

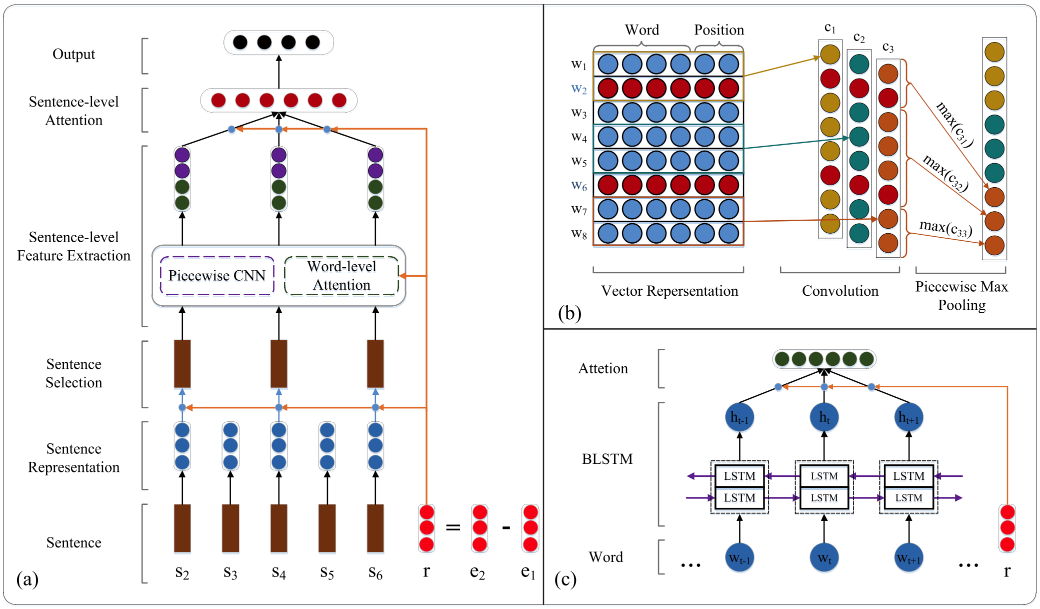


Fig. 1. An illustration of Hierarchical Selective Attention Network (HSAN). (a) The overall architecture of HSAN. (b) Piecewise Convolutional Neural Networks is utilized to capture sentence features, we utilize 3 filters, and the length of each filter is 2. (c) A word-level attention mechanism is used to capture sentence features.

3.1. Sentence representation

3.1.1. Word embeddings

Word embeddings are distributed representation of words that map each word to a low-dimensional real-valued vector, which captures syntactic and semantic meaning of the word. Given a sentence $S_i = \{w_1, w_2, \dots, w_n\}$ with two marked entities $e_1(= w_p)$ and $e_2(= w_q)$, $(p, q \in [1, n], p \neq q)$, each word w_i is transformed into a vector $\mathbf{w}_i^d$ by looking up pre-trained word embedding matrix $\mathbf{V} \in \mathbb{R}^{|V| \times d_w}$, where V is a fixed-sized vocabulary and d_w is the dimension of word embeddings.

Many knowledge graph embedding approaches (Bordes, Usunier, Garcia-Duran, Weston, & Yakhnenko, 2013; Lin, Liu, Zhu, Zhu, & Zhu, 2015; Wang, Zhang, Feng, & Chen, 2014) regarded relation as translation from head entity (e_1) to tail entity (e_2), i.e. $\mathbf{e}_1 + \mathbf{r} = \mathbf{e}_2$. $\mathbf{e}_1, \mathbf{r}, \mathbf{e}_2$ is the vector format of e_1, r, e_2, respectively. Specially, for a bag labelled by $r(e_1, e_2)$, the difference vector $\mathbf{r} = \mathbf{e}_2 - \mathbf{e}_1$ contains the features of relation r. Each instance in the bag

ïĄĎ may express the relation r or not. If an instance express the relation r, its feature vector should have higher similarity with $\mathbf{r}$, otherwise lower similarity. Thus we use $\mathbf{r} = \mathbf{e}_2 - \mathbf{e}_1$ to denote the relation vector of the two entities e_1 and e_2.

3.1.2. Position embeddings

In relation extraction, it is necessary to specify which input tokens are the target nouns in the sentence. Similar to Zeng, Liu, Lai, Zhou, and Zhao (2014), we use position features (PF) to specify entity pairs. The PF is defined as the combination of the relative distances from the current word to two corresponding entities e_1 and e_2. For example, the relative distances of "founded" in sentence S_1 to e_1 (*Microsoft*) and e_2 (*Bill Gates*) are -1 and 4, respectively. Every relative distance is mapped to a randomly initialized position vector in $\mathbb{R}^{d_p}$, where d_p is the position vector dimensionality. For a given word w_i, we obtain two position vectors $\mathbf{w}_{i,1}^p$ and $\mathbf{w}_{i,2}^p$ with regard to entities e_1 and e_2.

We concatenate the word vector $\mathbf{w}_i^d$ and two position vectors $\mathbf{w}_{i,1}^p$ and $\mathbf{w}_{i,2}^p$ to form a new representation of word w_i, i.e., $\mathbf{w}_i = [\mathbf{w}_i^d; \mathbf{w}_{i,1}^p; \mathbf{w}_{i,2}^p]$ ($\mathbf{w}_i$ is the new vector of w_i, and $[x_1; x_2; x_3]$ denotes the concatenation of x_1, x_2 and x_3). All the words in sentence S_i form a matrix $\mathbf{S}_i = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_n\} \in \mathbb{R}^{n \times d}$, where $d = d_w + d_p * 2$.

3.1.3. Sentence representation

In this section, we transform sentences into their representation by aggregating their word representations. We feed the matrix $\mathbf{S}_i = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_n\} \in \mathbb{R}^{n \times d}$ into a bag-of-words (BoW) model, which sums the resulting vectors as follows:

$$\mathbf{z}_i = \sum_{i=1}^{n} \mathbf{w}_i, \tag{1}$$

where $\mathbf{z}_i \in \mathbb{R}^d$.

3.2. Sentence selection

In our work, we narrow our search space and focus on reading only sentences that are likely to be relevant to the relation r. We use the following formulas as our selection mechanism.

$$\alpha_i = \frac{exp(\omega_i)}{\sum_{j=1}^{n} exp(\omega_j)}, \tag{2}$$

$$\omega_i = W_a(\tanh[\mathbf{z}_i; \mathbf{r}]) + b_a, \tag{3}$$

where $[x_1; x_2]$ represents the concatenation of x_1 and x_2, $W_a \in \mathbb{R}^{d+d_w}$ is a row vector, and b_a is an offset value. $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_l]$ is the weight vector of all instances in the bag B. Then we can select top m relevant sentences by sorting α from highest to lowest.

3.3. Sentence-level feature extraction

In relation extraction, an input sentence that is marked as containing the target entities corresponds only to one relation type rather than predicting labels for each word. Thus, it might be necessary to utilize all local features and predicting a relation globally. In this section, we use two different neural models to extract local features from the selected sentences of a bag. A word level attention mechanism base on Bidirectional Long Short-Term Memory Networks (BLSTM) is utilized to mining important word features of each sentence. And PCNN is used to mining word phase features and the structural information between two entities of each sentence.

3.3.1. Piecewise convolutional neural networks

PCNN has been shown to be effective for distant supervised relation extraction tasks (Lin et al., 2016; Zeng et al., 2015). A sentence is inherently divided into three segments according to the two given entities: one internal context and two external context, which involves the characters inside or around the two entities, respectively. As shown in Fig. 1(b), PCNN contains two parts: Convolution and Piecewise Max Pooling, which utilizes convolution operation to extract features from sentences and piecewise max pooling procedure to determine the maximum value in each segment based on the positions of the two given entities.

Convolution. A convolution operation involves a filter $\mathbf{w} \in \mathbb{R}^{h\times d}$, which is applied to a window of h words to produce a new feature. For example, a feature c_i is generated from a window of words $\mathbf{w}_{i:i+h-1}$ by

$$c_i = f(\mathbf{w}\cdot\mathbf{w}_{i:i+h-1} + b), \tag{4}$$

where $b \in \mathbb{R}$ is a bias term, f is a non-linear function such as the hyperbolic tangent, and out-of-range input values $\mathbf{w}_i$, where $i < 1$ or $i > n$, are taken to be zero. This filter is applied to each possible window of words in the sentence S_i to produce a feature map as follows:

$$\mathbf{c} = [c_1, c_2, \ldots, c_n], \tag{5}$$

where $\mathbf{c} \in \mathbb{R}^n$.

Generally, the convolution operation may contain multiple filters with varying window sizes to capture different features. Here, we use k filters with the window size of h, and then the convolution result is a matrix as follows:

$$\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k\}, \tag{6}$$

where $\mathbf{C} \in \mathbb{R}^{k\times n}$.

Piecewise max pooling. Single max pooling operation is often used to extract the most significant features in the feature maps $\mathbf{C}$. However, it reduces the size of the hidden layers too rapidly and cannot capture the structural information between two entities. In order to capture structural and other latent information, PCNN divides a sentence into three segments and extracts the maximum value in each segment. As shown in Fig. 1(b), the output of each convolutional filter $\mathbf{c}_i$ is divided into three segments $\{\mathbf{c}_{i1}, \mathbf{c}_{i2}, \mathbf{c}_{i3}\}$ by two entities. The piecewise max pooling procedure can be expressed as follows:

$$p_{ij} = max(\mathbf{c}_{ij}),\ 1 \le i \le n,\ 1 \le j \le 3. \tag{7}$$

Thus we can extract a 3-dimensional vector $\mathbf{p}_i = \{p_{i1}, p_{i2}, p_{i3}\}$ for the output of each convolutional filter. We then concatenate all vectors $\mathbf{p}_{1:k}$ and apply a non-linear function, such as the hyperbolic tangent. Finally, the piecewise max pooling operation outputs a vector as follows:

$$\mathbf{g}_{ic} = tanh(\mathbf{p}_{1:k}), \tag{8}$$

where $\mathbf{g}_{ic} \in \mathbb{R}^{3k}$ is a sentence representation of sentence S_i.

3.3.2. Word-level attention

Attentive neural methods have demonstrated success in a wide range of tasks ranging from question answering (Chen, Fisch, Weston, & Bordes, 2017), machine translation (Bahdanau, Cho, & Bengio, 2014), speech recognition (Chorowski, Bahdanau, Serdyuk, Cho, & Bengio, 2015), image captioning (You, Jin, Wang, Fang, & Luo, 2016). Important hidden vectors correspond to important parts in the sentence regarding to the generation of the word and contribute more to the formation of the sentence vector. In this section, we use a word-level attention mechanism to assign a weight to each hidden vector generated by BLSTM and construct a sentence representation.

Bidirectional LSTM. As shown in Fig. 1(c), we first utilize BLSTM to encode the hidden states of all the ordered words $\{w_1, w_2, \ldots, w_n\}$ in the sentence S_i and we set the number of hidden units in Long Short-Term Memory Networks (LSTM) equal to the dimension of the input word vector, i.e. d. At time-step t, the forward LSTM and the backward LSTM encode the word w_t as hidden states $\overrightarrow{\mathbf{h}}_t = \overrightarrow{LSTM}(\mathbf{w}_t)$ and $\overleftarrow{\mathbf{h}}_t = \overleftarrow{LSTM}(\mathbf{w}_t)$, respectively. Then we sum the forward hidden states and the backward hidden states, and the output of the word w_t is shown in the following equation:

$$\mathbf{h}_t = \overrightarrow{\mathbf{h}}_t \oplus \overleftarrow{\mathbf{h}}_t, \tag{9}$$

here, we use element-wise sum to combine the forward and backward pass outputs.

Attention. It is obvious that not all words contribute equally to the sentence meaning for different relations. Hence, instead of feeding hidden states to a Bow module, we adopt a word-level attention mechanism to extract specific words that are important to the meaning of a sentence. Finally, we aggregate the representation of those informative words to form the sentence representation. The sentence vector is computed as a weighted sum of these annotations $\mathbf{h}_i$ as follows:

$$\mathbf{g}_{ia} = \sum_{i=1}^{n} \beta_i \mathbf{h}_i, \tag{10}$$

where $\mathbf{g}_{ia}$ is also a sentence representation of sentence S_i, β_i measures the importance of the jth word for the relation r and it is computed by as follows:

$$\mu_i = \mathbf{W}_b(tanh[\mathbf{h}_i; \mathbf{r}]) + b_b, \tag{11}$$

$$\beta_i = \frac{exp(\mu_i)}{\sum_{j=1}^{n} exp(\mu_j)}, \tag{12}$$

where $\mathbf{W}_b \in \mathbb{R}^{d+d_w}$ is a row vector, and b_b is a bias value.

3.4. Sentence-level attention

For a sentence S_i in the bag B, we utilize PCNN and word-level attention to extract two different sentence representations $\mathbf{g}_{ic}$ and $\mathbf{g}_{ia}$. Then we concatenate $\mathbf{g}_{ic}$ and $\mathbf{g}_{ia}$ to form a new sentence representation $\mathbf{g}_i \in \mathbb{R}^{3k+d}$

$$\mathbf{g}_i = [\mathbf{g}_{ic}; \mathbf{g}_{ia}]. \tag{13}$$

We translate each sentences selected to a vector, and the result of the bag is a matrix $\mathbf{B} = \{\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_m\}$.

The method BoW we used in choosing relevant sentences is relatively simple, so the selected sentences may contain noise data. If we regard all the chosen sentences equally, the wrong labelling sentences will bring in massive of noise during training and testing. Hence, we use a sentence-level attention module to dynamically highlight the important instances in the selected sentences.

Similar to Section 3.2, we first compute the similarity of the selected sentence S_i and the relation r, namely, ν_i. The calculation procedure of ν_i is as follows:

$$\nu_i = W_c(\tanh[\mathbf{g}_i; \mathbf{r}]) + b_c, \tag{14}$$

where $W_c \in \mathbb{R}^{3k+2d}$ is a row vector, and b_c is a offset value. We then utilize a softmax function to obtain a normalized importance weight γ_i and denote it as:

$$\gamma_i = \frac{exp(\nu_i)}{\sum_{j=1}^{m} exp(\nu_j)}. \tag{15}$$

After that, we compute the bag vector $\mathbf{G}$ as a weighted sum of the selected sentences in the bag B based on the weights as follows:

$$\mathbf{G} = \sum_{i=1}^{m} \gamma_i \mathbf{g}_i. \tag{16}$$

3.5. Softmax output and objection function

Assume that there are N bags in training set $\{B_1, B_2, \ldots, B_N\}$, and their labels are relations $\{r_1, r_2, \ldots, r_N\}$. The bag vector $\mathbf{G}_i$ hierarchical extracted from the words and sentences of the bag B_i is a high level representation of the bag and can be used as features for relation classification. We use a non-linear layer to project $\mathbf{G}$ into the target space of $|y|$ class and feed it to a softmax classifier to predict the semantic relation label $\hat{y}$:

$$\hat{\mathbf{G}}_i = \tanh(\mathbf{W}_d \mathbf{G}_i + b_d), \tag{17}$$

$$\hat{y}_i = \frac{\exp(\hat{\mathbf{G}}_i)}{\sum_{j=1}^{|y|} exp(\hat{\mathbf{G}}_j)}, \tag{18}$$

where $\mathbf{W}_d \in \mathbb{R}^{3k+2d}$ is a row vector, and b_d is a bias value.

A reasonable training objective to be minimized is the categorical cross-entropy loss, which is calculated as a regularized sum:

$$J(\theta) = -\frac{1}{|y|} \sum_{i=1}^{|y|} y_i \log(\hat{y}_i) + \lambda \, \|\theta\|_F^2, \tag{19}$$

where y_i is the gold probability and $\hat{y}_i$ is the predict probability of class i, λ is a L2 regularization hyper-parameter, and θ indicates all parameters of our model. To prevent over-fitting, we also employ dropout (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) on the output layer.

4. Experimental setup

In this section, we introduce the dataset, the evaluation metrics and the hyper-parameters used in this paper.

4.1. Dataset and evaluation metrics

We evaluate our model HSAN on a widely used dataset which is developed by Riedel et al. (2010). This dataset was generated by aligning Freebase with the New York Times (NYT) corpus. The dataset is tagged with the Stanford named entity tagger (Finkel, Grenager, & Manning, 2005) to find entity mentions, and then these entity mentions are matched to the names of Freebase entities. The training dataset and the testing dataset are the aligned sentences from NYT corpus of the years 2005–2006 and the year 2007 respectively. The dataset contains 53 relations (including no relation "NA") and 39,528 entities. The training contains 522,611 sentences, 281,270 entity pairs and 18,252 relational facts. The test dataset contains 172,448 sentences, 96,678 entity pairs and 1,950 relational facts. We use *word2vec*[1] to train word embedding on the NYT corpus and use the embeddings as initial values.

Following previous work (Ji et al., 2017; Lin et al., 2016; Mintz et al., 2009; Zeng et al., 2015), we evaluate our model HSAN in the held-out evaluation, which evaluates our model by comparing the extracted relation facts with those in Freebase, and report both the precision/recall curves and Precision@N (P@N) of the experiments.

Table 2
Effect of hyper-parameters.

	P	R	F1
100	0.37	0.305	0.334
150	0.358	0.344	0.351
200	0.389	0.325	0.354
230	0.358	0.363	0.36
250	0.367	0.358	**0.362**

4.2. Hyper-parameter settings

In this section, we experimentally study the effects of the hyper-parameters used in this paper. We use a grid search to determine the optimal parameters and select the dimension of word embedding d_w among $\{50, 60, \ldots, 300\}$, the position embedding dimension d_p among $\{5, 10, 15, 20\}$, the window size h among $\{2, 3, 4, 5, 6, 7\}$, the number of feature maps k among $\{100, 150, 200, 230, 250\}$, and the batch size among $\{50, 60, \ldots, 100\}$. We employ dropout and L2 regularization to prevent over-fitting and select their parameters among $\{0.1, 0.2, \ldots, 0.6\}$ and $\{0.00001, 0.0001, \ldots, 0.1\}$, respectively. We use AdaDelta (Zeiler, 2012) in the update procedure and select the learning rate among $\{1.0, 0.9, \ldots, 0.1\}$. The ρ and ϵ used in AdaDelta are 0.95 and $1e^{-6}$, respectively. Following previous methods (Ji et al., 2017; Lin et al., 2016; Zeng et al., 2015), we tune all the parameters using three-fold validation on the training set.

In Table 2, we respectively set the word embedding dimension, the position embedding dimension, the window size, the batch size, the dropout, L2 regularization and the learning rate as 50, 5, 3, 80, 0.1, 0.00001, 1.0, vary the number of feature maps and compute the F1. Table 2 shows that the F1 is growing, when using more feature maps, but it is growing slowly.

In Table 3, we show all parameters used in the experiments. As other parameters have little effect on the results, we set them the same as Lin et al. (2016).

5. Experimental results and analysis

In this section, we show the experimental results and comparisons with previous baselines.

5.1. Evaluation results

We compare our method with the following four previous works:

Mintz (Mintz et al., 2009) is a traditional distant supervised model.

MultiR (Hoffmann et al., 2011) proposes a probabilistic, graphical model of multi-instance learning which handles overlapping relations.

MIML (Surdeanu et al., 2012) jointly models both multiple instances and multiple relations.

PCNN+ATT (Ji et al., 2017; Lin et al., 2016) employs PCNN to construct sentence representation of each sentence in a bag and then automatically assign weights to these sentence representations by utilizing sentence-level attention (ATT) to form the bag representation.

We implement them with the source codes released by the authors which achieve comparable results as the authors reported. We also implement another two neural network models:

Se+PCNN+ATT first selects the relevant sentences from a bag, then employs PCNN and sentence-level attention to construct the bag representation based on these selected sentences.

PCNN+ATT+WA utilizes not only PCNN but also word-level attention (WA) to extract important features from all the sentences

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

Table 3
Parameters used in this paper.

Word dimension	Position dimension	Window size	Feature maps
50	5	3	230
Batch size	Dropout probability	L2 regularization	Learning rate
80	0.1	0.00001	1.0

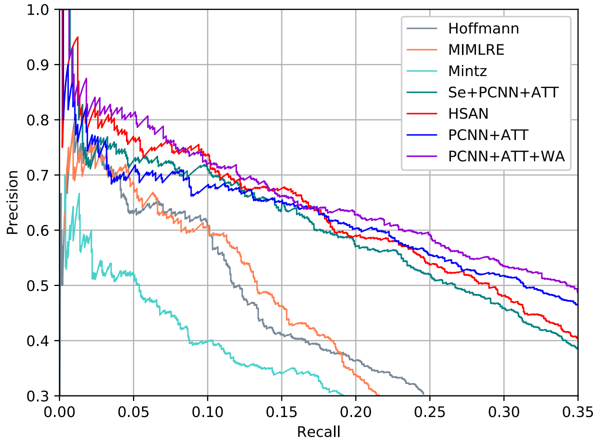


Fig. 2. Precision–recall curves of our methods against traditional methods.

Table 4
P@N of our models and time consuming of each bag. The best two values in each column are marked in bold.

Test settings	P@N (%)				Time (ms)	
	100	200	300	Mean	Train	Test
Se+PCNN+ATT	75.0	70.5	64.67	70.06	**9.54**	**2.76**
HSAN	**79.0**	**73.5**	**67.67**	**73.39**	**17.55**	**3.67**
PCNN+ATT	75.0	70.4	66.67	70.69	49.97	7.12
PCNN+ATT+WA	**80.0**	**73.5**	**68.67**	**74.06**	91.56	14.69

Table 5
Performance on our model HSAN with different number of selected sentences. HSAN is our default model, which first selects 10 sentences from each bag.

Test settings	P (%)	R (%)	F1 (%)
HSAN	39.18	33.31	36.00
HSAN-20	37.26	35.78	36.50
HSAN-30	36.54	37.92	37.22
HSAN-40	34.68	43.15	38.45
HSAN-50	33.49	49.53	**39.96**

in a bag and employs sentence-level attention to select important instances from a bag.

Fig. 2 displays the precision–recall curves for each method. Fig. 2 indicates that:

1. HSAN significantly outperforms all feature-base methods (Hoffmann et al., 2011; Mintz et al., 2009; Surdeanu et al., 2012) over the entire range of recall. When the recall is greater than 0.1, the performance of feature-based methods drop out quickly, while HSAN has a reasonable precision until the recall approximately reaches 0.3. It demonstrates that the human-designed feature cannot concisely express the semantic meaning of the sentences, and the inevitable error brought by NLP tools will hurt the performance of relation extraction. Automatically learning features via neural networks can alleviate the error propagation that occurs in traditional feature extraction.
2. HSAN achieves better performance than Se+PCNN+ATT, PCNN+ATT+WA achieves better performance than PCNN+ATT, which both indicate that word-level attention mechanism can filter out meaningless words and select important words to form the sentence representation, which indirectly alleviates the wrong labelling problem.
3. Compared with PCNN+ATT and PCNN+ATT+WA, HSAN performs much better when the recall is low (almost 0.2), we can conclude that selecting relevant sentences from a bag helps filter out meaningless sentences. As the recall increases, the precision gradually decreases and falls faster, the reason is that we only use a small amount of information by selecting 10 relevant sentences from tens of thousands of instances, while PCNN+ATT and PCNN+ATT+WA use tens of thousands of sentences, thus we spend less time, and the results are not that good.

5.2. P@N metrics and time consuming

In this section we use P@N metrics to evaluate our models and count the time consuming of each bag on test dataset.

Following Lin et al. (2016), we rank the predictions according to the confidence scores given by our models, and then count the precision of the top N samples. We do experiments on a single GPU device Tesla K80 and count the training and testing time of our model on each bag. The result is shown in Table 4. It is founded that:

1. HSAN performs better than Se+PCNN+ATT, and PCNN+ATT+WA performs better than PCNN+ATT, which again demonstrate that the word-level attention module is effective to highlight specific words that are important to the sentence meaning. However, during training and testing, Se+PCNN+ATT runs 1.84 and 1.33 faster than HSAN, PCNN+ATT runs 1.83 and 2.06 faster than PCNN+ATT+WA, which indicate that the word-level attention module is time consuming. Since we utilize BLSTM module to learn word representation before the word-level attention module, and BLSTM module processes a word at each time, thus it is time consuming.
2. In P@100, P@200, P@300, HSAN not only gets higher precision PCNN+ATT, but also runs 2.85 and 1.94 faster than PCNN+ATT during training and testing respectively. This is because we first employ coarse sentence-level attention to select most relevant sentences, which can filter out meaningless sentences, and utilize these selected sentences to predict the relation.
3. In P@100, P@200, P@300, HSAN gets similar precision as PCNN+ATT+WA, in addition, HSAN runs 5.22 and 4.00 faster than PCNN+ATT+WA during training and testing respectively, which also demonstrate the effect of the coarse sentence-level attention.

5.3. Effect of sentence number

In this section, we select different number of sentences by taking coarse sentence-level attention on all sentences in a bag to demonstrate the effect of our model HSAN.

Table 5 depicts the performance of our model HSAN on different number of selected sentences. HSAN, HSAN-20, HSAN-30, HSAN-40, HSAN-50 represents 10, 20, 30, 40, 50 sentences are selected

Table 6
Some examples of selective attention in NYT corpus.

Relation	Sentence
/business/company/founders	In January, [YouTube]'s **co-founder**, [Chad Hurley], said the company would in the coming months begin sharing advertising revenue with contributors.
/people/person/place_of_death	[Michael Dibdin], an internationally acclaimed British crime novelist whose best-known books feature the brooding Italian police detective Aurelio Zen, **died** on March 30 in [Seattle].
/location/country/capital	Take Vienna's florid architecture, throw in Budapest's bubbling cafe culture, and you get [Zagreb], [Croatia]'s grand **capital**.
/people/person/place_of_birth	[Michael Smuin], the son of a Safeway butcher, was **born** in [Missoula], Mont, on Oct. 13, 1938.

respectively, in each bag during training and testing. Table 5 indicates that:

1. The F1 score of HSAN gets better when more sentences are retrieved. Especially, compared with HSAN, HSAN-50 boosts the F1 score by 5.27%.
2. The recall gradually increases when selecting more sentences. Since we utilize coarse sentence-level attention to retrieve more sentences from a bag, and then the fine sentence-level attention aggregates these selected sentences to construct the bag representation. Thus the bag representation contains more details about the two entities e_1 and e_2, so HSAN can predict the real relation between e_1 and e_2 or "NA" if there is no relation.
3. The precision gradually decreases when selecting more sentences. This is because Freebase is not complete, and the predicted true relation instances may be misclassified.

5.4. Case study

Table 6 shows some examples of word-level attention in test dataset. We use italic and bold to represent entity and keyword respectively. Our models assign high weights to the keywords and low weights to other words in a sentence. Table 6 indicates that the words with high weights is helpful to predict relation. For example, **co-founder** helps to make sure that the relation between *Chad Hurley* and *YouTube* is */business/company/founders*. The other three examples also illustrate this phenomenon. Therefore, the word-level attention mechanism can select the keywords and is useful in our work.

6. Conclusions

In this paper, we propose a novel hierarchical selective attention network to extract relations from texts under distant supervision. Since an entity pair maybe appears in tens of thousands instances, to reduce the computational cost, we first employ coarse sentence-level attention to select most relevant instances and use these selected instances to predict the relation. Considering words are not the same important to the meaning of a sentence, we employ a word-level attention mechanism to extract such words that are important to the sentence meaning and aggregate them to form a sentence representation. Experimental results show that our method outperforms most of existing state-of-the-art methods.

Acknowledgment

This work is supported by the National High Technology Research and Development Program of China (863 Program) (Grant No. 2015AA015402), the National Natural Science Foundation of China (No. 61602479 and 61702514) and the Strategic Priority Research Program of the Chinese Academy of Sciences (XDBS01070000).

References

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473.

Bengio, Y., Schwenk, H., Sencal, J. S., Morin, F., & Gauvain, J. L. (2006). Neural probabilistic language models. *Journal of Machine Learning Research (JMLR)*, *3*(6), 1137–1155.

Bian, J., Liu, Y., Agichtein, E., & Zha, H. (2008). Finding the right facts in the crowd:factoid question answering over social media. In *International conference on world wide web (WWW)* (pp. 467–476).

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. *Advances in Neural Information Processing Systems*, 2787–2795.

Chen, D., Fisch, A., Weston, J., & Bordes, A. (2017). Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th annual meeting of the association for computational linguistics (ACL)* (pp. 1870–1879).

Chorowski, J. K., Bahdanau, D., Serdyuk, D., Cho, K., & Bengio, Y. (2015). Attention-based models for speech recognition. In *Advances in neural information processing systems* (pp. 577–585).

Dietterich, T. G., Lathrop, R. H., & Lozanoperez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, *89*, 31–71.

Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., et al. (2014). Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In *ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 601–610).

Finkel, J. R., Grenager, T., & Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Meeting on association for computational linguistics (ACL)* (pp. 363–370).

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A., Jaitly, N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(6), 82–97.

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

Ji, G., Liu, K., He, S., & Zhao, J. (2017). Distant supervision for relation extraction with sentence-level attention and entity descriptions. In *Proceedings of the thirty-first AAAI conference on artificial intelligence* (pp. 3060–3066).

Jiang, X., Wang, Q., Li, P., & Wang, B. (2016). Relation extraction with multi-instance multi-label convolutional neural networks. In *Proceedings of the 26th international conference on computational linguistics (COLING)* (pp. 1471–1480).

Lin, Y., Liu, Z., Zhu, X., Zhu, X., & Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence* (pp. 2181–2187).

Lin, Y., Shen, S., Liu, Z., Luan, H., & Sun, M. (2016). Neural relation extraction with selective attention over instances. In *Meeting of the association for computational linguistics (ACL)* (pp. 2124–2133).

Mintz, M., Bills, S., Snow, R., & Dan, J. (2009). Distant supervision for relation extraction without labeled data. In *Joint conference of the meeting of the acl and the international joint conference on natural language processing of the AFNLP (ACL-IJCNLP)* (pp. 1003–1011).

Qu, J., Ouyang, D., Hua, W., Ye, Y., & Li, X. (2018). Distant supervision for neural relation extraction integrated with word attention and property features. *Neural Networks*, *100*, 59–69.

Riedel, S., Yao, L., & Mccallum, A. (2010). Modeling relations and their mentions without labeled text. In *European conference on machine learning and knowledge discovery in databases* (pp. 148–163).

Sa, C. D., Ratner, A., Christopher, R., Shin, J., Wang, F., Wu, S., et al. (2016). Deepdive: declarative knowledge base construction. *Sigmod Record*, *45*(1), 60–67.

Schmidhuber, J., Meier, U., & Ciresan, D. (2012). Multi-column deep neural networks for image classification. *IEEE conference on computer vision and pattern recognition (CVPR)*, *157*(10), 3642–3649.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR)*, *15*(1), 1929–1958.

Sun, H., Ma, H., Yih, W. T., Tsai, C. T., Liu, J., & Chang, M. W. (2015). Open domain question answering via semantic enrichment. In *International conference on world wide web (WWW)* (pp. 1045–1055).

Surdeanu, M., Tibshirani, J., Nallapati, R., & Manning, C. D. (2012). Multi-instance multi-label learning for relation extraction. In *Joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)* (pp. 455–465).

Wang, Z., Zhang, J., Feng, J., & Chen, Z. (2014). Knowledge graph embedding by translating on hyperplanes. In *Twenty-eighth AAAI conference on artificial intelligence* (pp. 1112–1119).

You, Q., Jin, H., Wang, Z., Fang, C., & Luo, J. (2016). Image captioning with semantic attention. In *Computer vision and pattern recognition (CVPR)* (pp. 4651–4659).

Zeiler, M. D. (2012). ADADELTA: an adaptive learning rate method, arXiv Preprint arXiv:1212.5701.

Zeng, D., Liu, K., Chen, Y., & Zhao, J. (2015). Distant supervision for relation extraction via piecewise convolutional neural networks. In *Conference on empirical methods in natural language processing (EMNLP)* (pp. 1753–1762).

Zeng, D., Liu, K., Lai, S., Zhou, G., & Zhao, J. (2014). Relation classification via convolutional deep neural network. In *Proceedings of the 25th international conference on computational linguistics (COLING)* (pp. 2335–2344).

Zeng, W., Lin, Y., Liu, Z., & Sun, M. (2017). Incorporating relation paths in neural relation extraction. In *Proceedings of the 2017 conference on empirical methods in natural language processing (EMNLP)* (pp. 1768–1777).

Zeng, D., Zeng, J., & Dai, Y. (2017). Using cost-sensitive ranking loss to improve distant supervised relation extraction. In *The 16th China national conference on computational linguistics (CCL)*.