Relation Extraction using Language Model Based on Knowledge Graph

Chengli Xing¹, Xueyang Liu², Dongdong Du³⁎, Wenhui Hu² and Minghui Zhang⁴

¹SiChuan Tianfu Bank Co., Ltd., China
²National Engineering Research Centre for Software Engineering, Peking University
³China Academy of Industrial Internet, China
⁴Handan Institute of Innovation, Peking University, China

⁎Corresponding author email: dudongdong@china-aii.com

Abstract. Relation extraction is an important task in natural language processing (NLP). The existing methods generally pay more attention on extracting textual semantic information from text, but ignore the relation contextual information from existed relations in datasets, which is very important for the performance of relation extraction task. In this paper, we represent each individual entity as a embedding based on entities and relations knowledge graph, which encodes the relation contextual information between the given entity pairs and relations. Besides, inspired by the impressive performance of language models recently, we used the language model to leverage word semantic information, in which word semantic information can be better captured than word embedding. The experimental results on SemEval2010 Task 8 dataset showed that the F1-score of our proposed method improved nearly 3% compared with the previous methods.

Keywords: Relation extraction, Knowledge graph, Language model.

1. Introduction

Information extraction is one of tasks in natural language processing (NLP). It mainly aims to extract structural information that can be dealing with by machine from any text documents. In recent years, the number of text information (or documents) on the internet grows quickly, which leads information extraction become one of the research hotspots. The information extraction roughly composed of three parts: relation extraction and classification, named entity recognition (NER) and event classification. As for relation extraction and classification, when dealing with relation classification, one needs to which pair of entities in a given sentence has relation. As for the relation extraction task, relation labels of the extracted relations need to be predicted. In this paper, we are focusing on the relation extraction tasks.

In the past of several years, deep learning had made a great development in NLP, and relation extraction is one of related fields that had big achievements. For examples, [22] achieved a great result by employing convolutional neural network(CNN) to extract features from word level and sentence level without importing any NLP Tools. [3] is based on CNN and replaced it with a more new effective loss function, so that it could be better to recognize the differences between different relationships. [4] learned nodes on the syntax tree through recurrent neural network(RNN), and finally obtained the vector representation of sentence used for relationship classification. [5] merged RNN with CNN and reached a great result.
In most NLP tasks, using word embedding to represent the input text is very common. With the help of pre-trained word embedding matrix, word embedding transforms a word into a real-valued vector, which captures the semantic meanings of the word. Word embedding are usually learned from large unlabeled text corpora, such as word2vec [2] or glove [1]. Using word embedding in most NLP tasks to represent the input raw text data can get desirable results. And its values can adjust during most cases. How-ever, those methods have many drawbacks, for example, the representation of each word is static, lose the information of context. Although we can adjust these embedding during downstream tasks, this fine-tuning work has little effect on the final result. Recently, pre-trained models with additional layers are shown to be effectiveness for many downstream tasks, such as ELMo [12] and BERT [13], which all focus on learning general word representations or sentence representations. Through a large number of unlabeled corpus, ELMo uses bidirectional LSTMs to generate contextual features. As for BERT, large transformers are employed on very large text corpora using a slightly modified masked language modeling objective. Both of them can get state-of-the-art word embedding. In our proposed method, we used BERT to encode the input sentences, and get high quality representations for the downstream tasks.

In relation extraction task, some researchers distinguish whether a word is an entity or not by using position features, which is the very basic representation of entities. Obviously, this method is extremely not enough for mining the information of entity. [7] employed entity vectors to solve this problem, which are random initialized, and can be optimized during the training process. In order to obtain more accurate and rich entity representation, [8] built an independent CNN-based model to get entity vectors using the background knowledge from Freebase and Wikipedia, which get obviously improvements. Nevertheless, this method needed lots of additional work to get properly entity vectors from additional knowledge. [9] analyzed sentence into syntax tree, then the sub syntax trees which using entity as their root are selected as entity representation. However, the methods mentioned previous ignored the inherent relations within different entities and correlation between entities and relationships. Thus, the quality of entity vectors usually stay poor. To deal with this problem, in this paper, we propose knowledge graph based on the entities and relations to obtain better entity representation.

First, we build an entities and relations knowledge graph (KG) [14]. KG represents domain knowledge and contains various nodes and directed edges. In this paper, the entities are the nodes in KG, and relations between entities are directed edges. Then we build TransR [17] based on KG to gain the embedding of nodes and edges. In graph embedding, relationship prediction is a common task, and it aims to predict what relationship exists between two node in a given knowledge graph. Usually, we compute two nodes to gain the potential relationship vectors between them, and compare it with all relationship vectors. The nearest relationship vector is the prediction result. We believe that the potential relationship vectors represent all possible relationship between two entities. The entity and relation embedding from TransR can represent the inherent relations within different entities and correlation between entities and relationships. And adding those embedding into our models can greatly enrich the information where relation classification tasks needed, find the inherent relations between different entities. For example, Jobs was the CEO in Apple Company, and now Cook take over his position in Apple. Thus, Jobs and Apple belongs to employment relationship; and the relationship between Jobs and Cook is colleague. When both Apple and Cook appear in one sentence at same time, the sentences is likely to express an employment relationship. With adding entity vectors, our model can learned this hidden rule, and help improve the final result of classifying.

To sum up, our contributions are mainly made up of three parts:

1) For relation extraction task, we employed both word embedding and language model to get the sentence representation. For the same word in different context, language model is able to get different feature vectors, leading a better sentence representations compared with word2vec or glove.

2) We get the entity and relation embedding based on entities and relations knowledge graph, which can learn the inherent correlation relatedness between entities and relations. Knowledge graph embedding learning from the entity and relation knowledge graph enrich the representation of entities and relations.
In the experiments, we showed that the performance of our proposed method KGRNN improved nearly 3% compared with other methods. The rest of the paper is organized as follows. Section 2 gives the motivation of our proposed method. Section 3 gives the details of our proposed model. While Section 4 provides the comparison experiment results using various algorithms. Section 5 gives the review of related works. In the end, Section 6 draws conclusions of this paper.

2. Motivation

2.1. Word Semantic Information cannot be Fully Captured via Traditional Word Embedding

It is crucial for NLP tasks to have a high-quality contextual information. Under different sentences, the same word may have different meanings. For example, in Table 1, the word “about” has different meanings in different sentences. For word “about”, if we just give the same word embedding to our model, the model is hard to understand different meanings in different contexts. That is, traditional word embedding methods lack contextual information, which leads the information contained in the word embedding is not comprehensive, thus affecting the performances of downstream tasks. Recently, using language model to extract fixed feature vectors have made impressive performances in many NLP tasks. BERT is a kind of language model, which can obtain pre-training language representations, it trains a language model on a large text corpus (like Wikipedia), and lots of NLP tasks reach better performances by employing BERT. For a same word, BERT will generate different feature vector based on different context, which can solve the problem mentioned above. So in relation extraction task, we also used BERT to extract feature vectors to get better representation of the input data.

2.2. Global Information of Relations is Helpful for Relation Extraction

As we all know, we can infer some hidden relationships between target entity pair using the existing relations. For example, in the training set, both Canada and Calgary have different types of relations between other entities, and Calgary has some relations with other entities, what we want to know is the relation type between entity pair (Canada, Calgary). In this data set, sentence1) and sentence2) are in the training set, and sentence3) is in the test set. As shown in the following, the entities in sentences are bolded.

1) When I was 4, we moved to north Alberta, Canada.
2) The land is near Calgary, while that is one of Alberta’s largest cities, the capital is Edmonton.
3) Some of the biggest creative names in Canada are choosing to introduce their new works in Calgary.

From the above instances and some other sentences in the training set, we build a entities and relations knowledge graph, which can be seen in Figure 1. The entities are the nodes in KG, and relations between entities are directed edges. The solid arrows are the relations that we already know, and dotted arrows are the missing relations that we want to predict. From sentence1) and sentence2), we can know that Alberta is a city belonging to Canada, and Calgary is Alberta’s largest city. So we can easily infer from KG that the relation between entity pair (Canada, Calgary) is contain.

With the example above, we can build TransR based on KG to gain the embedding of nodes and edges. These embedding can represent the inherent relations within different entities and correlation between entities and relationships. Using the entity and relation embedding, our proposed model can infer the relation type between target entity pairs more precisely.
Table 1. Examples about the word ‘about’.

| Contextual Sentence A                                                                 | Actual label            |
|---------------------------------------------------------------------------------------|-------------------------|
| The propaganda <e1>machine</e1> tells us about these tin-pot <e2>dictators</e2>, which are the greatest threat to the world. | Other                   |
| The most common <e1>audits</e1> were about <e2>waste</e2> and recycling.               | Message-Topic           |
| In its last 12 meetings this year the <e1>committee</e1> has made about 30-35 <e2>recommendations</e2>. | Product Producer        |
| Michael Jackson’s FBI <e1>file</e1> consists of about 600 <e2>pages</e2>.              | Component-Whole         |

Figure 1. A simple example of entity and relation knowledge graph.

3. Methodology
Given an entity pair \((e_i, e_j)\), and a sentence \(s = \{x_1, x_2, ..., x_n\}\) that contains these two entities, our proposed method can get the probability of each relation type \(r\). Figure 2 shows the overview of our proposed model. Using sentence encoder, we are able to get the representation of the input sentence, and using entity encoder to get entity embedding. Finally, combing the two parts and via a softmax layer, our model will get the probability of each relation type \(r\). Our method contains the following parts:

1) **Sentence Encoder.** For the input sentence \(s\) composed of \(n\) words \(\{x_1, x_2, ..., x_n\}\), word embedding, BERT and BiGRU are employed to encode the sentence and extract more useful and high-level information.

2) **Entity Encoder.** From the entity and relation knowledge graph, our model can get the entity and relation embeddings with the inherent correlation information.

3) **Classification Layer.** Combing the representations of sentence and entities, a softmax layer is employed to get the probability of each relation type \(r\).

3.1. **Sentence Encoder**
For the input sentence \(s\) composed of \(n\) words, we first transform the input raw text data into low-dimensional dense real-valued feature vectors using BERT. Furthermore, we introduce position features, which can identify the target entity and its relative position with other words. Then BiGRU is used to get the representation of input data, which gives our model the ability to get different contextual information. Finally, using max pooling to reduce the size of output and get the most important information.

1) **Input Representation:**
   a) **Word Feature Vectors.** For each sentence in the corpus, we first used BERT to get its feature vector. Due to the limitation of hardware, we only use BERT-Base model to represent the input sentences, extract the last layer’s output of BERT-Base model as the feature vector, the dimension of the feature vector is \(d^{w1}\), where \(d^{w1} = 768\).  


Next, we used word embedding to get more information, the dimension of word embedding is $d_{w2}$.

b) Position Embedding. For the relation extraction task, traditional word embedding can’t get the relative position information between entity and other input words, which is very important for the relation type extraction. Similar to [10], we introduce position embedding to encode the relative distance of each word from the two target entities in the sentence. With the help of position embedding, the model can learn how close each word is to each entity, and we believe that more useful information regarding the relation is hidden in the words closer to the target entities. Take sentence “Willie Morris grew up in Yazoo City” as an example, the entity pair is (Willie Morris, Yazoo City), relative distance of the word “grew” to head entity “Willie Morris” is 1 and to tail entity “Yazoo City” is −3, because there are two entities, so we need two position embedding to determine the relative position of two entities with other words. The relative distance values are then encoded in a $d_p$ dimensional vector.

Finally, after concatenating the word embedding and position embedding, we transform a sentence into a matrix $X = \{w_1, w_2, \ldots, w_n\}$ as the input representation, where $w_i \in \mathbb{R}^{d_{w1}+d_{w2}+2*d_p}$, and $n$ is the length of input sentence. The matrix $X$ is then fed into a BiGRU network.

2) BiGRU and Max-pooling Layer: BiGRU is employed to merge all embedding features to get the global information. GRU is first introduced by [8], which is a kind of advanced recurrent neural network (RNN), it can capture short-term and long-term dependencies. Because GRU employs an adaptive gating mechanism to adjust information, it is very suitable to deal with sequential data such as text data.

GRU has two kinds of gate units, which called update gate $z_t$ and reset gate $r_t$. The update gate $z_t$ is used to control how many information from the previous status can be kept at the current status. The larger value of the update gate, the more information from previous status is kept. The reset gate $r_t$ is used to control the degree of ignoring the information from previous status. The smaller value of the reset gate, the more information from previous status is ignored.

The $z_t$ and $r_t$ are calculated as follows:

$$r_t = \sigma(W_r x_t + U_r h_{t-1})$$  \hspace{1cm} (1)

$$z_t = \sigma(W_z x_t + U_z h_{t-1})$$  \hspace{1cm} (2)

---

**Figure 2.** The overview of our proposed model.
Where $\sigma$ is a sigmoid function. $x_t$ is the input information at time $t$. The hidden state $h_t$ is computed as follows:

$$h_t = \tanh(W_h x_t + U_h (r_t \ast h_{t-1}))$$  \hspace{1cm} (3)

Where $\ast$ denotes element-wise multiplication. $W_r, U_r, W_z, U_z, W_h$ and $U_h$ are all parameter matrices which are initialized randomly (or using some specified initialization methods), and will be adjusted during the training step. Finally, GRU will get a new status $h_t$:

$$h_t = (1 - z_t)h_{t-1} + z_t f_t$$ \hspace{1cm} (4)

Generally, we often use Bidirectional-GRU to get full information from both past and future context, which is composed of a forward GRU layer and a backward GRU layer. For a input sentence $s$ composed of $n$ words $\{x_1, x_2, ..., x_n\}$, the forward GRU layer will encode $s$ from $x_1$ to $x_n$, the result is denoted as $r_l$. Similarly, the backward GRU layer will encode $s$ from $x_n$ to $x_1$, which denoted as $r_r$. And the final output of Bi-GRU is the concatenate of $r_l$ and $r_r$, that is $R = [r_l, r_r]$.

Finally, max-pooling layer is used to reduce the size of output and maintain the most important information from the previous layer. After embedding layer, BiGRU layer, maxpooling and connect layer, the sentence $s$ is represented as $\mathcal{E}_{\mathcal{E}_{\mathcal{E}_{\mathcal{E}_{\mathcal{E}_{\mathcal{E}}}}}}$.

### 3.2. Entity Encoder

In this part, we will introduce how to get the entity and relation embedding from KG, which is the key part in our proposed method, it can obtain entity embedding and relation embedding for downstream tasks, such as relation extraction.

Given a corpus $C$ and an entity pair $(h, t)$ under the relation type we what to predict, with applying for our knowledge graph, all entities that appear in $C$ are nodes, all the known relations that appear in $C$ are the directed edges in knowledge graph, and the link between entity pair $(h, t)$ is dotted arrows which represents the missing relations we want to predict. Based on our definition, Figure 1 shows a simple example of knowledge graph.

In entity and relation knowledge graph, a triplet $(h, r, t)$ is composed of an entity and its relation, where $h$ and $t$ stand for the head and tail entity, respectively, and $r$ is the relation between the given entity pair. Usually, the entities embedding are set as $h, t \in \mathbb{R}^d$, and relation embedding is set as $t \in \mathbb{R}^k$, and the dimensions of entity embedding and relation embedding are not necessarily equal.

Relations and entities are completely different objects, it may be not capable to represent them in a common semantic space. To address this problem, [17] proposed a new method, which models entities and relations in distinct spaces, i.e., entity space and relation spaces, and performs translation in relation space, which called TransR. Figure 3 shows a simple example of TransR.

For each relation $r$, a projection matrix $M_r$ is set to project entities from entity space to relation space. Using projection matrix, we can define the projected vectors of entities as:

$$h_r = h M_r, \ t_r = t M_r$$ \hspace{1cm} (5)
To learn such embedding, we minimize a margin-based ranking criterion over the training set:

$$L = \sum_{(h,t,r) \in S} \sum_{(\hat{h},\hat{t},r) \in S} \left[ y + f_r(h, t) - f_r(\hat{h}, \hat{t}) \right]_+$$

(6)

Where $S$ is all triplets in the entity and relation knowledge graph, $\hat{S}$ is the negative triples sets generated from $(h, r, t)$, which are incorrect triples. $[x]_+$ is the positive part of $x$, i.e. $\max(0, x)$. $f_r(h, t)$ is the score function for $h$ and $t$ under relation $r$, which is defined as:

$$f_r(h, t) = \|h_r + r - t_r\|^2_2$$

(7)

The parameters of the entity embedding, relation embedding and projection matrix are initialized randomly, and will be adjusted during training with the help of stochastic gradient descent (SGD).

Based on entity and relation knowledge graph, we will get the embedding of target entities, denoted as $E_{Ent1}$ and $E_{Ent2}$.

### 3.3. Classification Layer

As show in Figure 2, after combing the sentence representation $Emb_{Sent}$ and entities embedding $Emb_{Ent1}$, $Emb_{Ent2}$, we use a softmax layer to get the probability of each relation type.

$$\text{softmax}([Emb_{Sent}, Emb_{Ent1}, Emb_{Ent2}])$$

(8)

### Table 2. Statistics of datasets.

| Relation          | #Train | #Test |
|-------------------|--------|------|
| Other             | 1410   | 454  |
| Effect-Cause      | 1003   | 328  |
| Whole-Component   | 941    | 312  |
| Entity-Destination| 845    | 292  |
| Producer-Product  | 717    | 231  |
| Entity-Origin     | 716    | 258  |
| Collection-Member | 690    | 233  |
| Message-Topic     | 634    | 261  |
| Agency-Instrument | 504    | 156  |
| Content-Container | 540    | 192  |
| Total             | 8000   | 2717 |

### Table 3. Hyperparamter settings.

| Hyperparameter | Value |
|----------------|-------|
| $d_{w1}$       | 768   |
| $d_{w2}$       | 300   |
| $d_{p}$        | 10    |
| $d_{mn}$       | 300   |
| $d_{Ent}$      | 20    |
| dropout_ratio  | 0.3   |

### 4. Experiment & Results

#### 4.1. Dataset

The SemEval-2010 Task 8 dataset is used to evaluate the performance of our KGRNN model. As show in Table 2, SemEval-2010 Task 8 dataset contains 8000 sentences for training, and 2717 for testing. There are 9 actual relation classes, together with an artificial class Other. Particularly, actual relations are directional, in other words, Effect-Cause(e1,e2) and Effect-Cause(e2,e1) are different relation class in our dataset. So we have $(2 * 9 + 1) = 19$ different classes for 10 relations.
4.2. Experiment Settings

We use BERT to extract fixed contextual representations of each word generated from the hidden layers of the pre-trained model. And we choose Google BERT-Base-Uncased model as our pre-trained model, which has 12 hidden layers and 768 dimension for each hidden layer. What’s more, we use the released word embedding set GoogleNews-vectorsnegative300.bin to initialize our embedding layer, which is trained by Mikolovs word2vec tool.

We use TransR as our graph embedding model to train the knowledge graph. We pre-train the entity embedding and use them directly in the subsequent model. The dimension of the entity embedding is 20.

Other hyperparameter settings are presented in Table 3.

4.3. Experiment Metrics

We use F1-score to verify the experiment performance in the validation process. These metrics are calculated as follows:

\[
Precision = \frac{TP}{TP + FP} \tag{9}
\]

\[
Recall = \frac{TP}{TP + FN} \tag{10}
\]

\[
F1-Measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{11}
\]

The variable definitions in the formula are as follows:

- TP: the Number of positive links correctly predicted.
- TN: the Number of negative links correctly predicted.
- FP: the Number of negative links incorrectly predicted.
- FN: the Number of positive links incorrectly predicted.

All baseline systems and our model use the official macro-averaged F1-score to evaluate model performance. This official measurement excludes the Other relation and takes directionality into account.

4.4. Experiment Results

To show the effectiveness of our KGRNN model, we compare our KGRNN model with other state-of-the-art methods. We select the SVM model proposed in (Rink and Harabagiu 2010), the CNN model proposed in (Zeng et al. 2014), the BRNN model proposed in (Zhang and Wang 2015), the CRCNN model proposed in (Santos et al. 2015) and the EAtt-BiGRU model proposed in (Qin et al. 2017) as our baseline model and implement them by ourselves which achieve comparable results as the authors reported.

As shown in Table 4, the SVM model, which achieves the best performance in traditional feature-based methods, extracted a large number of features from the sentence using existing NLP tools and achieved an F1-score of 82.21%. From the perspective of convolution, Zeng et al. (2014) constructed a CNN model on the word sequence; they also integrated word position embedding and lexical features to improve the performance of relation classification. The CNN model achieved F1-score of 82.21% with these information. BRNN used the original RNN and max-pooling operation to extract sentence-level feature; position indicator features are also used in BRNN to extract information focused on the entities.
Table 4. Comparison with previous relation classifications systems

| Model                                | Additional Information                                                                 | F1  |
|--------------------------------------|----------------------------------------------------------------------------------------|-----|
| SVM (Rink and Harabagiu 2010)        | POS, prefixes, morphological, WordNet, dependency, parse, Levin classed, ProBank, FrameNet, NomLex-Plus, Google ngram, paraphrases, TextRunner | 82.21 |
| CNN (Zeng et al. 2014)               | Word embedding + PF                                                                      | 82.21 |
| BRNN (Zhang and Wang 2015)           | Word embedding + PI                                                                      | 83.34 |
| CR-CNN (Santos et al. 2015) W        | Word embedding + PF                                                                      | 81.90 |
| EAtt-BiGRU (Qin et al. 2017)         | Word embedding + PF                                                                      | 83.18 |
| KGRNN W                              | Word feature vectors + Entity embedding                                                  | 85.28 |
| Collection-Member                    | 690                                                                                    | 233 |
| Message-Topic                        | 634                                                                                    | 261 |
| Agency-Instrument                    | 504                                                                                    | 156 |
| Content-Container                    | 540                                                                                    | 192 |
| Total                                | 8000                                                                                   | 2717 |

Based on above, it achieved an F1-score of 83.34%. CR-CNN focused more on the influence of class Other, which proposes a new pairwise ranking function to substitute softmax. This targeted modification obtains F1-score of 81.90%. EAtt-BiGRU used attention mechanism to leverage information from entity embedding, which achieved F1-score of 83.18%. We make use of two types of information to improve the performance of KGRNN: Word Semantic features and Entity relation features. Our proposed KGRNN model yields an F1-score of 85.28%, outperforming existing competing approaches. According to the experiment results, KGRNN can effectively accomplish the relation classification task.

5. Related Work

5.1. Relation Extraction

Relation classification is an important topic in NLP. Classical relation classification methods are based on labeled data set, which is composed of handcrafted features and kernel methods [5]. The classifiers rely on the handcrafted features, such as POS information, and other features, Levin classed and n-gram features [20]. Those features are domain-specific and data-specific. Different features provide different performance, even if we use the same features, different data may resulting in different performance, so the generalization ability of handcrafted features is very weak. Additionally, it’s very time-consuming and need lots of professional knowledge. Kernel methods employ existed NLP tools, which can transform the input into parse tree. However, it also has some drawbacks, for example, those NLP tools may make some mistakes through their processing, and the downstream tasks will inherit the mistakes, finally will cause the bad influence on the model’s performance.

Recently, there are more and more deep learning models for relation extraction. [21] first introduced to use a CNN to learn features automatically without hand-craft features. It first encodes the input sentence, and a convolutional kernel layer is followed, finally, using a single layer full-connected layer and a softmax output to get the final prediction. Similar to the previous model, [22] used CNN for relation extraction, but this model used word embedding which obtained from a large unlabeled corpus, besides, this paper first introduced position embedding, which improves the performance a lot. [23] completely discarded hand-craft features, which incorporated convolutional kernels of varying windows sizes to capture the required features, and found that using kernels with 2-3-4- 5 window length can get best performance. [24] proposed a model can Piecewise Convolutional Neural Networks(PCNN), which used the multi-instances learning paradigm, and the most important contribution of this paper is piecewise maxpooling.

Generally speaking, the max-pooling layer can drastically reduce the size of output and maintain the most important information from the previous layer. However, it also not sufficient to capture the structure between the entities in the sentence. PCNN adopts piecewise max pooling in relation
extraction to avoid the problem. In PCNN, each convolutional filter $pi$ is divided into three segments $(pi_1; pi_2; pi_3)$ by head and tail entities, and the max-pooling operation is used in three segments separately. [25] used an attention mechanism over all the sentences in the bag for the multi-instance problem.

The results showed that this selective attention mechanism can significantly improve the model’s performance. [26] used LSTM and tree structures for relation extraction task. This model is composed of three parts: embedding layer, sequence layer and dependency layer. The embedding is used to encode the input sentence, sequence layer can identify whether a word is an entity or not, and the dependency layer is designed for relation extraction. [6] used BiLSTM and attention mechanism to improve the model’s performance. [27] proposed a novel Hierarchical attention-based Bidirectional Gated recurrent neural network integrated with entity Descriptions (denoted by HBGD), which can address the problem of wrong labels and obtain the most useful semantic information.

5.2. Knowledge Graph Embedding

Knowledge Graph Embedding is first introduced by Google [28]. Knowledge graph usually transforms a structured data as the form of triplets $(h, r, t)$, where $h$ is the head entity, $t$ is the tail entity, and $r$ is the relation between the given entities. However, the numerical machine learning methods cannot be learned from the traditional knowledge which is composed of symbolic and logic [19]. To solve this problem, some researchers project entities and relations into embedding by employing knowledge graph embedding. [15] TranE is the first model which used embedding method. The basic idea is the relation between two entities corresponds to a translation between the embedding of entities. TransE applies well to 1-to-1 relations but has issues when applied to N-to-1, 1-to-N and N-to-N relations. TransH [16] is proposed to get distinct distributed representations when involved in different relations. Where TransR [17] models entities and relations in distinct spaces, i.e., entity space and multiple relation spaces, and performs translation in the corresponding relation space.

6. Conclusion

To tackle the classification problem, we propose a novel approach KGRNN using the Language Model based on Knowledge Graph. With the help of Language Model, the word semantic information can be better captured. And on top of semantic information from text, KGRNN learns the embedding of entities and relations via knowledge graph and thus adds to the model the semantics of relation context. The experimental results on SemEval-2010 relation classification task demonstrate that KGRNN has better performance than that of the state-of-the-art method in terms of F1-measure.

Acknowledgments

This work is supported by National Key R&D Program of China (No. 2017YFB1103602, No. 2017YFB1103003).

References

[1] J. Pennington, R. Socher, C.D. Manning, Title of paper with only first word capitalized, EMNLP 2014: 1532-1543.
[2] Mikolov, Tomas, Chen, Kai, Corrado, et al., Efficient Estimation of Word Representations in Vector Space Computer Science, 2013.
[3] Ccero Nogueira dos Santos, Bing Xiang, Bowen Zhou: Classifying Relations by Ranking with Convolutional Neural Networks. ACL (1) 2015: 626-634
[4] Richard Socher, Brody Huval, Christopher D. Manning, Andrew Y. Ng: Semantic Compositionality through Recursive Matrix-Vector Spaces. EMNLP-CoNLL 2012: 1201-1211
[5] Rui Cai, Xiaodong Zhang, Houfeng Wang: Bidirectional Recurrent Convolutional Neural Network for Relation Classification. ACL (1) 2016
[6] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, Bo Xu: Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. ACL (2) 2016
[7] Pengda Qin, Weiran Xu, Jun Guo: Designing an adaptive attention mechanism for relation classification. IJCNN 2017: 4356-4362
[8] Guoliang Ji, Kang Liu, Shizhu He, Jun Zhao: Distant Supervision for Relation Extraction with Sentence-Level Attention and Entity Descriptions. AAAI 2017: 3060-3066
[9] Zhengqiu He, Wenliang Chen, Zhenghua Li, Meishan Zhang, Wei Zhang, Min Zhang: SEE: Syntax-Aware Entity Embedding for Neural Relation Extraction. AAAI 2018: 5795-5802
[10] Wenpeng Yin, Hinrich Schütze, Bing Xiang, Bowen Zhou: ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. TACL 4: 2016 259-272
[11] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, Eduard H. Hovy: Hierarchical Attention Networks for Document Classification. HLT-NAACL 2016: 1480-1489
[12] Peters, Matthew, M. Neumann, et al., Deep Contextualized Word Representations. NAACL-HLT 2018: 2227-2237.
[13] Devlin, Jacob and Chang, et al., BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805.
[14] Q. Liu, Y. Li, H. Duan, Y. Liu, Z. Qin, Knowledge graph construction techniques, in Journal of Computer Research and Development, 2016, 53(3): 582-600. (in Chinese)
[15] A. Bordes, N. Usunier, A. García-Durán, J. Weston, O. Yakhnenko, Translating Embeddings for Modeling Multi-relational Data, In NIPS 2013: 2787-2795.
[16] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge Graph Embedding by Translating on Hyperplanes, In AAAI 2014: 1112-1119.
[17] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning Entity and Relation Embeddings for Knowledge Graph Completion, In AAAI 2015: 2181-2187.
[18] G. Ji, S. He, L. Xu, K. Liu, J. Zhao, Knowledge Graph Embedding via Dynamic Mapping Matrix, In ACL 2015: 687-696.
[19] H. Xiao, M. Huang, X. Zhu, TransG : A Generative Model for Knowledge Graph Embedding, In ACL 2016.
[20] B. Rink, S.M. Harabagiu, UTD: Classifying Semantic Relations by Combining Lexical and Semantic Resources, SemEval@ACL 2010: 256-259.
[21] ChunYang Liu, WenBo Sun, WenHan Chao, and Wanxiang Che. Convolution neural network for relation extraction, In International Conference on Advanced Data Mining and Applications 2013. Springer, pages 2312.
[22] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. Relation classification via convolutional deep neural network, In COLING. 2014, pages 23352344.
[23] Thien Huu Nguyen and Ralph Grishman Relation extraction: Perspective from convolutional neural networks, In Proceedings of NAACL-HLT 2015, pages 3948.
[24] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. Distant supervision for relation extraction via piecewise convolutional neural networks, In EMNLP, pages 17531762.
[25] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. Neural relation extraction with selective attention over instances, In Proceedings of ACL. volume 1, pages 21242133.
[26] M. Miwa, M. Bansal, End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures, ACL (1) 2016.
[27] H. She, B. Wu, B. Wang, R. Chi, Distant Supervision for Relation Extraction with Hierarchical Attention and Entity Descriptions, IJCNN 2018: 1-8.
[28] Singhal, Amit, Introducing the knowledge graph: things, not strings, In Official google blog 5 (2012).