Relationship classification based on dependency parsing and the pretraining model

Baosheng Yin¹,² · Yifei Sun¹,²

Accepted: 23 April 2022 / Published online: 4 June 2022
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract
As an important part of information extraction, relationship extraction aims to extract the relationships between given entities from natural language text. Based on the pretraining model R-BERT, this paper proposes an entity relationship extraction method that integrates an entity dependency path and pretraining model, which generates a dependency parse tree by dependency parsing, obtains the dependency path of an entity pair via a given entity, and uses an entity dependency path to exclude information such as modifier chunks and useless entities in sentences. This model has achieved good F1 value performance on the SemEval2010 Task 8 dataset. Experiments on datasets show that dependency parsing can provide context information for models and improve performance.

Keywords Relationship extraction · Dependency parsing · Pretraining model

1 Introduction
Information extraction (IE) aims to extract structured information from large-scale semistructured or unstructured natural language text (Golshan et al. 2003). Information extraction tasks are applied, for example, to knowledge graph construction (Liu et al. 2016), information retrieval (Wu and Weld 2010), question-answering systems and text summarization. Entity relationship extraction is an important part of information extraction tasks, and its results will affect the performance of follow-up tasks.

Entity relationship extraction based on deep learning falls into one of the following two major categories: supervised entity relationship extraction and distantly supervised entity relationship extraction. In supervised entity relationship extraction, entity relationship extraction can be achieved by either pipeline learning or joint learning (Li et al. 2020). The pipeline learning method extracts the relationships between entities directly based on entity identification, and the joint learning method identifies entities while extracting the relationships between entities based on an end-to-end neural network model. Compared with supervised entity relationship extraction, distantly supervised entity relationship extraction, due to the lack of a human-annotated dataset, takes one more step to distantly align the knowledge base to label unlabeled data. To construct the relationship extraction model, there is little difference between distantly supervised entity relationship extraction and the pipeline learning method of the supervised entity relationship extraction. The main difference between supervised entity relationship extraction and distantly supervised entity relationship extraction is the difference in the annotation level of the dataset. For the supervised method, entity and relationship types are given in the dataset. At this time, the relationship extraction task can be performed by the classification task method.

Due to the increase in the pretraining models, the entity relationship extraction tasks have gradually moved closer to the direction of pretraining models. Researchers have achieved very good results by only fine-tuning the pretraining model BERT and then performing entity relationship extraction experiments. In 2019, the Ali team (Wu and He 2019) took the lead in applying BERT to relationship extraction tasks and achieved the best results at that time. The result led to more researchers focusing on

---
¹ Research Center for Human-Computer Intelligence, Shenyang 110136, China
² Shenyang Aerospace University, Shenyang 110136, China
the pretraining model. After these experiments, most entity relationship extraction models have been based on pre-training models, usually by training their pretraining models after changing the initialization parameters of the BERT structure or integrating external knowledge.

Peng et al. proposed in 2020 that the existing models paid too much attention to the impact of the whole sentence on the relationship classification, without considering the noise caused by content such as modifier chunks in sentences. Moreover, external knowledge is cited to assist the model in sentence classification, but the syntactic knowledge of the sentence is ignored.

This paper proposes a method using dependency parsing, which establishes a dependency tree for each data point and obtains the shortest dependency path between entities via dependency trees. This paper focuses on the word information in the dependency path between entities rather than using the type of dependency relationship between words in the past. Parsing is used to enhance the context information of model learning to avoid noise caused by information such as modifier chunks and unannotated entities in sentences.

2 Related work

Parsing is one of the key technologies in natural language processing, and its basic task is to determine the syntactic structure of a sentence or to clarify the dependency relationships between words in a sentence. Dependency parsing analyzes a sentence into a dependency syntax tree to describe the dependency relationships between words. The dependency relationship is represented by a directed arc, which is called the dependency arc. The shortest dependency path refers to the shortest path of two words in the dependency syntactic structure. The entity dependency path is the shortest path between two entity nodes in the dependency syntactic structure. The shortest dependency path can express the syntactic relationships between two nodes. According to the characteristics of the shortest dependency path, the entity dependency path can concisely express the syntactic relationships between entities, remove modifier chunks, and retain the backbone mode that can clearly express the entity relationships. Therefore, dependency parsing is widely used in relationship extraction.

Entity relationship extraction, as one of the most critical tasks in natural language processing, is widely used in fields such as information extraction, natural language understanding, and information retrieval. Early relationship extraction methods include feature-based methods and kernel-based methods. As early as feature-based methods, syntactic knowledge was used for relationship extraction. Today’s relationship extraction methods can be divided into two categories: statistical relationship extraction and neural relationship extraction. The statistical relationship extraction annotates the relationships of the target entity pair in the text based on traditional machine learning methods. Among them, classical entity relationship extraction methods can be divided into four categories: supervised, semisupervised, weakly supervised and unsupervised methods, which are distinguished by the degree of annotation in the dataset. Neural relationship extraction applies deep learning to relationship extraction tasks, and the entity relationship extraction tasks of deep learning can be divided into supervised tasks and distantly supervised tasks.

Among the classical statistical relationship extraction methods, Zhou (Zhou et al. 2005) and Guo Xiyue et al. (Guo et al. 2014) used SVM as a classifier to study the effects of lexical, syntactic and semantic features on entity relationship extraction. Craven et al. (Craven and Kumlien 1999) first proposed the idea of weakly supervised machine learning in the process of extracting structured data from text to establish a biological knowledge base; Brin (Brin 1998) used the bootstrapping method to extract the relationships between named entities. Hasegawa et al. (Hasegawa et al. 2004) first proposed an unsupervised method for extracting relationships between named entities at the ACL meeting.

Traditional methods have the error propagation problem of feature extraction, so the entity relationship extraction method based on deep learning, which can effectively solve this problem, has been considered and achieved good results. Zeng et al. (Zeng et al. 2014) first proposed using a CNN to extract the meaning of a word and applying softmax for classification in 2014. Zhang et al. (Zhang and Wang 1508) proposed using Bi-LSTM for relationship classification in 2015. Xu et al. (Xu et al. 2015) re-introduced the traditional method and proposed a CNN that is based on the shortest path. In addition, in recent years, many attention-based models have been applied to relationship extraction tasks. Katiyar et al. (Katiyar and Cardie 2017) first used attention, an attention mechanism, and Bi-LSTM to jointly extract entity and classification relationships in 2017.

Scholars have also proposed a variety of improvements based on basic methods, such as the fusion method of PCNN and multi-instance learning (Zeng et al. 2015) and the PCNN fusion method and attention mechanism (Lin et al. 2016). Ji et al. (Ji et al. 2017) proposed adding entity description information based on PCNN and attention to assist in learning entity representations. The COTYPE model proposed by Ren et al. (Ren et al. 2016) and the residual network proposed by Huang (Huang and Wang 2017) both enhanced the effect of relationship extraction.
After the pretraining model was proposed, Wu et al. first applied the pretraining model to relationship extraction tasks in 2019 and explored the mode of combining entities and entity locations in the pretraining model by adding identifiers before and after the entities to indicate the entity locations rather than using the traditional location vector. The best results were achieved at that time, which led more researchers to focus on the pretraining model. Later, Livio Baldini Soares et al. (Soares et al. 2019) from the Google team proposed a pretraining model of BERTEM + MTB in 2019. In that paper, the effects of input and output on the results of relationship classification under different conditions were discussed, and matching the blank pretraining task was proposed according to the results to eliminate the error caused by the overutilization of entities. Peng et al. (Peng et al. 2020) conducted experiments based on BERT and MTB in 2020, explored the information types used by the existing models in the entity relationship extraction tasks, designed experiments, and finally concluded that the existing models did not make full use of context information. Chen et al. proposed a new pretraining model in 2021 that integrated entity type information during pretraining, conducted experiments on multiple relational classification datasets, and achieved good results on small sample datasets. In this paper, the BERT model is used for experiments. In addition to the entity information, the entity dependency path is used as a syntactic representation, and sentence information, entity information and syntactic information are used as sentence representations for relationship classification.

3 Model introduction

In supervised entity relationship extraction tasks, since the dataset has fully annotated entities and the corresponding relationship types are given, the existing models all use these tasks as classification tasks. The model outputs a vector as a sentence representation and predicts the relationship type. This paper proposes a model framework that integrates entity type information during pretraining, and its basic task is to determine the syntactic structure of a sentence or the dependency relationships between words in a sentence. The dependency syntax was first proposed by French linguist L. Tesniere (1959) in his works, which analyzed a sentence into a dependency syntax tree to describe the dependency relationships between words. In the structure of dependency grammars, there is a direct dependency relationship between words to form a dependency pair, one of which is the core word, also known as the governing word, and the other is called the modifier, also known as the dependent word. The dependency relationship is represented by a directed arc, called the dependency arc. Take the sentence “The < e1 > show < /e1 > centered around a < e2 > beach theme < /e2 > ” as an example for dependency parsing. The analyzed dependency relationship of the sentence is shown in Fig. 2:

To obtain the entity dependency path, the dependency tree should be obtained from the dependency structure of the sentence first. According to the dependency tree and the annotated entities, the path between entities e1 and e2 on the dependency tree can be found, which is the entity dependency path. The entity dependency path is shown in Fig. 3, where the red nodes represent entity nodes, and the dotted line is the entity dependency path.

3.2 Input

The pretraining model BERT is a multilayer two-way transformer encoder. The input to BERT can be a sentence or a pair of sentences. A special tag [CLS] is the first tag of each sequence.

Given a sentence S, the dependency parsing tree is obtained through dependency parsing, and the shortest dependency path between entities is found according to the target entities (e1, e2). To prevent the path length from being 0 because one entity is dependent on the other entity, the entity dependency path is a path that contains the entities. Special identifiers are inserted before and after the two target entities to emphasize the entities and assist the model in capturing the locations of the entities.

The processed sentences and the entity dependency path are entered into the model. The location of the node words can be obtained by the entity dependency path, and the input of the one-hot vector is entered in the path. A [CLS] tag is added to the beginning of the sentence, and the data are input into a tokenizer to obtain its token sequence. The
vector representation of each token is generated by the encoder.

### 3.3 Sentence representation

The hidden state sequence $H$ output by the BERT module corresponds to each token. $H_0$ is the hidden state vector corresponding to the [CLS] token, for which an activation operation is performed, the result is input to a fully connected layer, and the resulting vector $H_0'$ is used as the representation of the sentence vector.

$$H = ENC(T)$$

$$H'_0 = W_0(\tanh(H_0)) + b_0$$

The hidden state vectors of the tags of the two pairs of entities are represented by $H_i$, $H_j$, $H_m$ and $H_n$. For the vectors of the target entities, the vector between $H_i$ and $H_j$ represents the vector of entity $e_1$, and the vector between $H_m$ and $H_n$ represents the vector of entity $e_2$. These vectors are summed and averaged to obtain a single representation, for which a tanh activation is performed, and the result is input to a fully connected layer to obtain the required entity vector representation.

$$H'^1 = W_1\left[\tanh\left(\frac{1}{n-m+1} \sum_{i=m}^{n} H_i\right)\right] + b_1$$

$$H'^2 = W_2\left[\tanh\left(\frac{1}{n-i+1} \sum_{i=1}^{n} H_i\right)\right] + b_2$$

For the vector representation of dependency syntax and the entity vector representation, the same method is adopted. POS represents the position of the word in the sentence. According to the shortest dependent path, we can obtain the position of the word on the path in the sentence. Then, tanh activation is performed, and the result is input to a fully connected layer to obtain a single syntax vector representation.
\[ p = \text{pos} \left( \text{word}_{\text{SDP}} \right) \]  

\[ H_{\text{SDP}} = W_3 \left[ \tanh \left( \text{Average} \left( \sum_{s \in p} H_s \right) \right) \right] + b_3 \ (s \in p) \]  

After the single representations of the sentence vector, entity vector and dependency syntactic vector are obtained, the four vectors are spliced to obtain a vector \( z \). The vector \( z \) is input to a fully connected layer, and the resulting vector is the final sentence representation \( r \) used for classification.

\[ r = W_4 \left[ \text{concat} \left( H_0, H^{e_1}, H^{e_2}, H_{\text{SDP}} \right) \right] + b_4 \]  

### 3.4 Classification

Given a sentence \( x \) containing the entire sentence and the analyzed shortest dependency path sequence, a vector representation \( r \) can be obtained by inputting \( x \) to a relational encoder. After the relationship representation is obtained, a fully connected softmax layer is used to predict the relationships of the sentence. Then, a probability distribution \( P \) covering all predefined relationship types is obtained.

\[ p(y|x, \theta) = \text{softmax}(W^r r + b^r) \]

where \( y \) is the target relationship type, and \( \theta \) refers to all learnable parameters, including \( W^r \) and \( b^r \).

### 4 Experiment

#### 4.1 Dataset

In this experiment, the SemEval-2010 Task 8 dataset was used. The dataset was collected from major data sources according to nine preset incompatible relationships, which contains 10,717 pieces of data, including 8,000 use cases for training and 2,717 use cases for testing. All examples in the dataset were annotated with nine relationships and an other relationship. The distribution of the nine relationship types is shown in Table 1.

In addition to the annotated relationship types, each data point also contains two annotated entities \( e_1 \) and \( e_2 \). The relationship types other than the other type are directional. For example, cause-effect \((e_1, e_2)\) and cause-effect \((e_2, e_1)\) are different. Therefore, in the experiments, (Table 2) 19 relationship types are usually set to make predictions.

In this paper, the macro-average F1 value in the official scoring script provided by the SemEval-2010 Task 8 dataset was used for scoring. According to this scheme, the macro-average F1 value scores of 9 actual relationships (excluding relationships of other types) were calculated, and the directionality of the relationships was considered. The calculation of F1 values requires precision and recall. The calculation formula is shown in Eqs. (9) to (11):

\[ \text{precision} = \frac{TP}{TP + FP} \]

\[ \text{recall} = \frac{TP}{TP + FN} \]

\[ F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \]

where true positive (TP) represents the number of correct predictions in positive prediction cases, false-positive (FP) represents the number of incorrect predictions in positive prediction cases, and false negative (FN) represents the number of incorrect predictions in the negative prediction cases.

### 4.2 Hyperparameter settings

The hyperparameters settings are as follows:

Hyperparameters were set to better compare with the baseline model (Wu and He 2019). Therefore, most of the hyperparameters were set to be the parameters of the

| Parameter        | Value |
|------------------|-------|
| Adam_epsilon     | 1e-8  |
| Learning_rate    | 2e-5  |
| Dropout_rate     | 0.1   |
| Train_epoch      | 5     |
| Batch_size       | 8     |
| Seed             | 42    |
| Max_sequence_length | 384  |

### Table 1 Relationship distribution of the SemEval-2010 Task 8 dataset

| Relationship type       | Training set | Testing set |
|-------------------------|--------------|-------------|
| Instrument-Agency       | 504          | 156         |
| Content-Container       | 540          | 192         |
| Message-Topic           | 634          | 231         |
| Member-Collection       | 690          | 233         |
| Entity-Origin           | 716          | 258         |
| Product-Producer        | 717          | 261         |
| Entity-Destination      | 845          | 292         |
| Component-Whole         | 941          | 312         |
| Cause-Effect            | 1,003        | 328         |
| Other                   | 1,410        | 454         |
baseline model, except for Batch_size and Max_sequence_length. Due to the hardware equipment, we used the smaller batch size for training and calculated the length of sentences that had been divided into tokens as the maximum sentence length.

4.3 Comparison of experimental results

Table 3 compares the performance of the model in this paper with various neural network models on the SemEval-2010 Task 8 dataset, which proves that the method proposed in this paper achieved good results. The highest value in each column of indicators is shown in bold.

It can be seen from the results in the table that the effect of the pretraining model is much better than those of neural network models such as CNN and LSTM. In this paper, the pretraining model was also used for experiments, and the R-BERT model was selected as the baseline model. The R-BERT model was based on the pretraining model and highlighted the entity information with special identifiers to indicate the entity location, which achieved the best results at that time, and the official F1 evaluation value reached 89.25%. On this basis, the shortest dependency path was obtained through dependency parsing and integrated into the R-BERT model in this paper so that the model could learn the context information of sentences. The results show that the F1 value performance of the model reached 89.97% after parsing was introduced, which fully proves that the context information provided by dependency parsing is effective.

We believe that the context information provided by the entity dependency path would play a large role, and a dependency syntax tree could be obtained after dependency parsing, which could provide much context information. We incorporated one of these paths, the dependency path between entities, into the pretraining model to provide the model with entity-related context information. The auxiliary model provided better prediction results. We also performed ablation experiments to prove the effectiveness of the entity-dependent path.

4.4 Ablation experiments

The method proposed in this paper was proven by the above experimental results. We wanted to further understand what factors, in addition to BERT, contributed to the experimental results in the method based on the pretraining model, and therefore, three ablation experiments were designed. Since the entity tags “<e1>” and “<e2>” were added to emphasize the entity and add boundary information to the entity, which significantly improved the classification prediction, these entity tags were reserved and used in each ablation experiment.

In the first experiment, a [CLS] token was added before the sentence input, the hidden layer vector of this token was used as a vector representation of sentence classification, and only this vector was used for classification. In the second experiment, [CLS] and the hidden vector of the entity dependency path were spliced to obtain a vector as the sentence representation, in which the entity dependency path did not contain entity information. In the third experiment, [CLS] and the hidden vector of the entity were spliced as sentence representation, and in this case, the entity information contained the tags of the entity and integrated the boundary information of the entity. The SDP represents the shortest dependency path. In the fourth experiment, we added SDP, which is the method we proposed based on the third experiment. After obtaining the SDP, we calculated statistics on the positions of all nodes on this path, summed all hidden vectors of nodes and took the average value. The obtained vector was the dependency syntactic vector, which was spliced with the [CLS] vector and entity vector to obtain the final representation of the sentence.

It can be seen from the results in Table 4 that the experimental results improved after the addition of entity identifiers, which provided the model with the boundary information of the entity and emphasize the entity. There

| Relationship Representation | F1/% |
|-----------------------------|------|
| [CLS]                       | 87.99|
| [CLS] + SDP                 | 89.15|
| [CLS] + ENT                 | 89.23|
| [CLS] + ENT + SDP           | 89.97|
was little difference between the result of using the hidden vector of entity dependency path information as sentence representation and that of using the hidden vector of entity as sentence representation, but the result of using entity information was better. Experimental results show that the model can make use of context information, but the model still needs entity information for supplementation. After combining the entity information with the context information provided by the dependency parsing, the model can predict the classification better.

### 4.5 Case study

This section analyzes the results of the R-BERT model and the model proposed in this paper in detail and compares the results of various relationship types, as shown in Table 5.

The results in the table show that the classification effect for most relationship types is improved compared with the baseline after introducing the entity dependency path, and the effect was more obvious for such relationship types as content-container, product-producer and instrument-agency, indicating that this experiment successfully integrated the entity dependency path into the pretraining model and was beneficial to improving the effect of relationship classification.

However, the classification effect for cause-effect and entity-destination did not improve but was significantly reduced. Therefore, we reviewed the classification results obtained by the two models in detail and extracted examples of incorrect classification results of the two models. Table 6 provides detailed examples of classification errors in these two types.

From the data classification results in the table, we can see that in the prediction results of these two types, the model proposed in this paper correctly predicted the relationship types but mispredicted the relationship directions, and the relationship types predicted by the baseline model were different from the standard results. Therefore, taking cause-effect as an example, when the accuracy of this type is calculated on the premise that the recall rates of the two models are not much different, due to the incorrect relationship directions in the prediction results on some data, the model in this paper predicted more data to be of cause-effect type than that of the baseline model, so the accuracy rate obtained was lower. As a result, the F1 value evaluation of the cause-effect classification results is lower than that of the baseline model.

Through the analysis of these two misclassified relationships, the length of the entity dependency path was 0 in the dependency syntax tree obtained by dependency parsing; that is, one entity depended on the other entity. Therefore, these cases did not obtain enough context information from the dependency syntax tree but only used the entity information again, leading to the correct prediction of relationship types and the incorrect prediction of relationship directions in some classification results.

It can be seen from the above results that the method proposed by the model in this paper not only allows the model to learn the context information provided by the dependency syntax but also improves the prediction of the

### Table 5 Comparison of F1 values of various relationship types

| Relation                  | R-BERT | R-BERT + SDP |
|---------------------------|--------|--------------|
|                           | P      | R    | F1   | P    | R    | F1   |
| Other                     | 72.05  | 57.39| 64.22| 71.86| 63.00| 67.14|
| Instrument-Agent          | 80.61  | 85.26| 82.87| 85.43| 82.69| 84.04|
| Component-Whole           | 86.25  | 88.46| 87.34| 87.30| 88.14| 87.72|
| Member-Collection         | 86.01  | 89.70| 87.82| 84.98| 92.27| 88.48|
| Entity-Origin             | 87.64  | 90.70| 89.14| 89.15| 89.15| 89.15|
| Product-Producer          | 85.31  | 90.48| 87.82| 90.39| 89.61| 90.00|
| Message-Topic             | 87.54  | 94.25| 90.77| 88.21| 94.64| 91.31|
| Cause-Effect              | 91.47  | 94.82| 93.11| 91.37| 93.60| 92.47|
| Content-Container         | 88.44  | 91.67| 90.03| 90.20| 95.83| 92.93|
| Entity-Destination        | 94.18  | 94.18| 94.18| 93.52| 93.84| 93.68|
| Official Score            | 87.49  | 91.06| 89.23| 88.95| 91.09| 89.97|

### Table 6 Comparative examples of results generated by models

| Prediction                          | Official | R-BERT | R-BERT + SDP |
|-------------------------------------|----------|--------|---------------|
|                                     | Entity-Destination(e2, e1) | Other | Entity-Destination(e1, e2) |
| Each time a neuron unleashes its tiny jolt, it needs to replenish its stores of energy for the next spark | Official | R-BERT | R-BERT + SDP |
|                                     | Cause-Effect(e1, e2) | Other | Cause-Effect(e2, e1) |
| These wind turbines generate electricity from naturally occurring wind | Official | R-BERT | R-BERT + SDP |
|                                     | Cause-Effect(e1, e2) | Content-Container(e1, e2) | Cause-Effect(e2, e1) |

A few days before the service, Tom Burris had thrown into Karen’s casket his wedding ring.

These wind turbines generate electricity from naturally occurring wind.
model. However, the model underutilizes the context information of the data in some relationship types, resulting in the incorrect classification of relationship types and incorrect classification of relationship directions. This case shows that there is still room for improvement in the use of context information, which is also the focus of our following work.

5 Conclusion

Based on the pretraining model, this paper proposed a pretraining model integrating dependency parsing for supervised entity relationship extraction. The shortest dependency path between entities is obtained by dependency parsing, which concisely expresses the syntactic relationships between entities, retains the main part of the expression relationship type, and removes useless modifier chunks and redundant entity information. The context information between entities can be obtained through dependency parsing, and a syntactic representation can be obtained by adopting the same processing method as the entity representation in the R-BERT model, which is spliced to the sentence vector and entity vector to obtain the vector representation for classification. The F1 value increased to 89.97% on the SemEval-2010 Task 8 dataset, which is an increase of 0.72% compared with R-BERT. Through the analysis of the results and the comparison of the results of the models, the model in this paper achieved good results, successfully learned the context information of sentences, basically solved the problems raised and achieved the expected results.

In the detailed analysis of the results, it was found that the length of the dependency path between entities in some sentences is 0. To avoid this situation, this paper also counts entities as nodes on the path during data processing. However, the model cannot obtain enough context information from the data, and the entity information is reused, which leads to the situation that in some relationship types, the entity relationship is accurately predicted, while the direction of the relationship is incorrectly predicted, affecting the final relationship prediction results to some extent. Therefore, in future work, we will attempt to design strategies to extract context information for these sentences to improve the overall effect of relationship extraction.

In this experiment, we only used the dependency paths between entities, but the whole dependency tree and types of dependency relationships were not effectively utilized. In addition, excluding all information outside the entity dependency path may also be the reason for the poor classification results of some relationship types. Therefore, our next goal is to make effective use of the entire dependency syntax tree and types of dependency relationships to obtain more complete context information to help the model judge the relationship types. In addition, for the dependency tree, we try the GCN model to obtain the information contained in the tree structure. This is because the GCN has a strong ability to obtain the information of the surrounding nodes. To avoid performance degradation due to noise, we plan to incorporate types of dependency relationships during training to selectively obtain information about the surrounding nodes.

Author contribution Author have made equal contribution.

Funding This work was supported by the National Defense Science and Technology Industrial Technology Research Project (JSQB2017206C002).

Availability of data and material Data for this work were obtained from the web (accessed from www.semeval2.fbk.eu/semeval2.php).

Declarations

Conflicts of interest The authors declare no conflict of interest.

References

Brin S (1998) Extracting Patterns and Relations from the World Wide Web[J]. Springer, Berlin Heidelberg, pp 172–183
Craven M, Kumlien J. Constructing Biological Knowledge Bases by Extracting Information from Text Sources[J]. Proceedings / International Conference on Intelligent Systems for Molecular Biology; ISMB. International Conference on Intelligent Systems for Molecular Biology, 1999:77–86.
Golshan PN, Dashti HR, Azizi S. A study of recent contributions on information extraction. arXiv preprint arXiv:1803.05667, 2018.
Gu M, Gu Y, Luo W et al (2021) From text to graph: a general transition-based AMR parsing using neural network. Neural Comput and Applic 33:6009–6025
Guo XY, He TT, Hu XH, Chen QJ. Chinese Entity Relationship Extraction based on Syntactic and Semantic Features. Journal of Chinese Information Processing, 2014,183–189 (in Chinese with English abstract).
Hasegawa T, Sekine S, Grishman R. Discovering Relations among Named Entities from Large Corpora[C]. Proceedings of the Meeting on Association for Computational Linguistics, 2004:415-es.
Huang YY, Wang WY. Deep Residual Learning for Weakly-Supervised Relation Extraction[C]. Proceedings of the Conference on EMNLP 2017: 1803–1807
Ji GL, Liu K, He SZ, et al. Distant supervision for relation extraction with sentence-level attention and entity descriptions[C]. In: Proc. of the AAAI. 2017. 3060–3066
Kattiyar A, Cardie C. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees[C]. Proceedings of the Conference on Association for Computational Linguistics. 2017. 917–928
Li DM, Zhang Y, Li DY et al (2020) Review of entity relation extraction methods[J]. J Comp Res Dev 7:1424–1448
Lin Y, Shen S, Liu Z, et al. Neural relation extraction with selective attention over instances[C]. In: Proc. of the Meeting of the Association for Computational Linguistics. 2016. 2124–2133
Liu Q, Li Y, Duan H, Liu Y, Qin ZG (2016) A survey of knowledge mapping construction techniques. J Comp Res Dev 53:582–600 ((in Chinese with English abstract))
Peng H, Gao T, Han X, et al. Learning from Context or Names? An Empirical Study on Neural Relation Extraction[C]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, November 16–20, 2020:3661–3672
Ren X, Wu Z, He W, et al. CoType: Joint Extraction of Typed Entities and Relations with Knowledge Bases[C]// the 26th International Conference. International World Wide Web Conferences Steering Committee, 2016:1015–1024
Soares L B, Fitzgerald N, Ling J, et al. Matching the Blanks: Distributional Similarity for Relation Learning[C]// Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019
Wang Q, Wang Y, Zhou Y et al (2022) SSPSNet: a single shot panoptic segmentation network for accurate scene parsing. Neural Comput and Appl 34:677–688
Wu SC, He YF. Enriching Pre-trained Language Model with Entity Information for Relation Classification[C]// the 28th ACM International Conference. ACM, 2019:2361–2364.

Wu F, Weld D S. Open information extraction using Wikipedia[C]// ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11–16, 2010:118–127
Xu K, Feng Y, Huang S, Zhao D (2015) Semantic relation classification via convolutional neural networks with simple negative sampling[J]. Comp Sci 71:941–949
Zeng D, Liu K, Lai S, Zhou G, Zhao J. Relation classification via convolutional deep neural network[C]. Proceedings of the 25th Int’l Conf. on Computational Linguistics: Technical Papers (COLING 2014). 2014. 2335–2344
Zeng D, Liu K, Chen Y, et al. Distant supervision for relation extraction via piecewise convolutional neural networks[C]. Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2015. 1753–1762.
Zhang D, Wang D. Relation classification via recurrent neural network. arXiv preprint arXiv:1508.01006, 2015.
Zhou GD, Su J, Zhang J, et al. Exploring various knowledge in relation extraction[C]. In: Proc. of the Conf. on Meeting of the Association for Computational Linguistics (ACL 2005). University of Michigan, 2002:419–444.

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.