Dependency Selection for Relation Triple Extraction

Zhiqiang Wang
Institute of Automation,
Chinese Academy of Sciences
University of
Chinese Academy of Sciences
wangzhiqiang2019@ia.ac.cn
Beijing, China

Yiping Yang*
Institute of Automation,
Chinese Academy of Sciences
yiping.yang@ia.ac.cn
Beijing, China

Junjie Ma
Department of Computer Science and Technology, Tsinghua University
junjiema@tsinghua.edu.cn
Beijing, China

Abstract

Recently, constructing an adjacency matrix through the dependency tree of a sentence and extracting relation triples via graph neural networks has proven to be successful. However, the issue of effectively pruning dependency trees has yet to be resolved. In this paper, we propose a method that utilizes dependency selection to address this problem. Specifically, we assign a weight to each dependency type and train the model. We then select the dependency type with the most significant weight changes to retain and discard the rest. Finally, we fine-tune the model with the retained dependency. Our experimental results on the public dataset SemEval demonstrate that our proposed method is effective and achieves competitive performance compared to the previous best model.

Keywords: relation extraction, knowledge graph, feature selection, graph neural networks

1 Introduction

Relation extraction (RE), which aims to identify entities in sentences and determine whether two entities in a sentence hold a semantic relation, plays an important role in many downstream tasks of natural language processing. For example, knowledge graph[1], automatic question and answering[2], sentiment analysis[3], etc.

In recent years, the method based on neural network has attracted more and more attention because of its automatic feature extraction and excellent performance[4, 5, 6]. Among them, the method based on graph neural networks has been widely developed in relation extraction tasks. However, how to prune the graph is still a challenge. [7] adopts the strategy of the shortest dependency path (SDP), and only retains the dependency relationship on the two entity connection paths. [8] argues that excessive pruning will ignore some semantic information. In [9], they adopt two types of pruning strategies, one is called Local and the other is Global.

In order to solve this problem, we propose a dependency selection method for relation extraction task. Concretely, we first train the model to select the dependency type with more

*Corresponding author.

drastic weight changes to retain, then, we fine-tune the model for the second time with the retained dependency. Experimental results on the public dataset SemEval show that our proposed method is effective and achieves competitive performance with the previous best model.

Our main contributions can be summarized as follows: (1) First, we use coarse-grained training to select dependencies and complete the pruning process. (2) Fine-tune the model with the retained dependencies to further improve the performance of the model. (3) A large number of experiments have been carried out on public data sets to fully verify the effectiveness of the model.

2 Related Work

AGGCN[10] is an early proponent of graph convolution network for relation extraction. In his method, firstly, he used the existing toolkit to analyze the dependency of sentences, and formed the adjacency matrix, which is used in graph convolution network. In addition, in order to capture semantic information of different granularity, the variable-length sublayer strategy is used to synthesize information through the dense connection layer. However, he did not distinguish the types of dependencies, that is, all dependencies are treated equally, which makes the performance of the model not high. A-GCN[9] pays attention to the type of dependency relationship. When calculating the adjacency matrix $\mathbf{A}$, the result derived by the type and two dependency words replace the traditional $\mathbf{A}$, and when calculating $\mathbf{A}$, the generalized attention mechanism was adopted. In the generated dependency matrix, four different pruning strategies are verified, however, When searching for the shortest path, a lot of calculations will be involved. [8] believes that pruning the dependency tree too hard would cause the negative meaning in the sentence to be deleted and the expressed information to be incomplete, which would damage the robustness of the model. [11] argues manual pruning strategy may lead to the omission of useful information. To solve this problem, the author proposes a dynamic pruning graph convolution network.

Our work is based on previous work, we don't prune dependency tree directly, but make a choice by observing the impact of each dependency type on the relation extraction task.

3 Our Proposed Method

The overview of our proposed model is shown in Figure 1(b). From which we can see that our model consists of four modules: encoder layer, graph neural networks layer/layers, entity representation layer and relation classification layer. Next, we will introduce each module in detail.

3.1 BERT Encoder

In this paper, we adopt BERT[12], a pre-trained language model, as our encoder. The encoding process is as follows

$$\{\mathbf{h}_1^0, \mathbf{h}_2^0, \cdots, \mathbf{h}_n^0\} = \mathrm{BERT}(\{x_1, x_2, \cdots, x_n\}) \tag{1}$$

where x_i denotes the i^{th} word in the sentence s, h_i^0 denotes the context representation of x_i and $h_i^0 \in \mathbb{R}^H$ $(0 \leq i \leq n)$.

3.2 Graph Neural Networks

Given a graph $\mathbf{G}$ with n nodes, we can use an $n \times n$ adjacency matrix $\mathbf{A}$ to represent the relationship between any two nodes i and j, that is, if there is a relationship between node i and node j, then $a_{i,j} = a_{j,i} = 1$, otherwise $a_{i,j} = a_{j,i} = 0$. Based on $\mathbf{A}$, the graph neural networks (GCN) can be denoted as

$$\mathbf{h}_i^l = \sigma(\sum_{j=1}^{n} a_{i,j}(\mathbf{W}^l \mathbf{h}_j^{l-1} + \mathbf{b}^l)) \tag{2}$$

where $\mathbf{h}_j^{l-1}$ is the context representation of x_j at $(l-1)^{\text{th}}$ layer, $\mathbf{W}^l$ and $\mathbf{b}^l$ are trainable matrix and vector at l^{th} layer, σ denotes activation function, such as ReLU. GCN can be regarded as node i in l^{th} layer use its adjacency nodes information represent itself. It is worth noting that the GCN layer can be stacked with several layers.

In our work, in order to select a subset from all dependency type, we use matrix $\mathbf{W}$ instead of matrix $\mathbf{A}$. Suppose $\mathcal{D} = \{d_1, d_2, \cdots, d_m\}$ denotes the dependency type set. Each d_i is a dependency type, such as $nsubj$, We construct

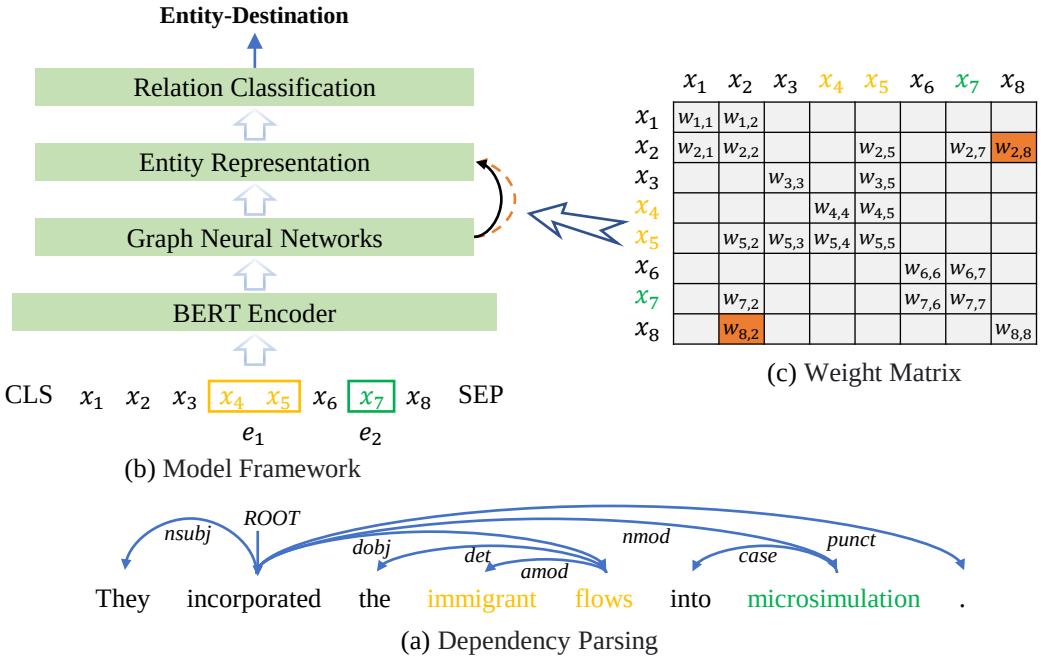

Figure 1: The overview of our model. (a) is an example sentence and its dependency parsing. (b) is the framework of our model. (c) is the adjacency matrix composed of the real values corresponding to the dependency type. The yellow and green parts in the sentence indicate the two entities in the sentence, respectively

a real value set $\mathcal{W} = \{w_1, w_2, \cdots, w_m\}$ corresponding to $\mathcal{D}$. So, each dependency type d_i corresponds to a real number w_i. In Figure 1(c), every $w_{i,j} \in \mathcal{W}$. In this way, if two nodes i and j hold dependency d_k, then $w_{i,j} = w_{j,i} = w_k$. Every real value w_i in $\mathcal{W}$ is trainable.

3.3 Entity Representation

After the operation of graph neural network, the representation of words in sentences has been fully interactive. In order to obtain the representation of entities in sentences, we adopt MaxPool operation for the word vector corresponding to the entities in sentences, as follows

$$\mathbf{h}_{e_k} = \text{MaxPool}(\{\mathbf{h}_i | x_i \in e_k\}) \quad (3)$$

where e_k denotes the k^{th} entity,$k = 1, 2$, $\mathbf{h}_i$ from the last layer of GCN is the context representation of x_i, $x_i \in e_k$ denotes x_i is a token in e_k.

Similarly, for the representation of sentences, we MaxPool the representation of words in the whole sentence.

$$\mathbf{h}_s = \text{MaxPool}(\{\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_n\}) \quad (4)$$

In this way, we get the representation of entities and sentences, respectively.

3.4 Relation Classification

After we get the representation of entities and sentences, we first concatenate them into one vector, then we feed them into a feed-forward neural network (FFNN) for information mixing. This can be expressed as follows

$$\mathbf{y} = \mathbf{W}_o(\mathbf{h}_s \oplus \mathbf{h}_{e_1} \oplus \mathbf{h}_{e_2}) + \mathbf{b}_o \quad (5)$$

Where, $\oplus$ denotes the concatenation operation, $\mathbf{W}$ and $\mathbf{b}_o$ are trainable matrix and vector.

Then, the softmax operation is adopted to the above results to determine the semantic relation

$$\hat{r} = \arg\max_r \frac{\exp(\mathbf{y}_r)}{\sum_{i=1}^{|\mathcal{R}|} \exp(\mathbf{y}_i)} \quad (6)$$

where $\mathcal{R}$ denotes the relation set and $\mathbf{y}_r$ is the t^{th} value of vector $\mathbf{y}$. We regard the relation category corresponding to the index with the highest probability as the final semantic relationship.

3.5 Training Objective

The training objective is defined as the sum of the cross entropy loss

$$J(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{r=1}^{|\mathcal{R}|}\mathbf{p}_r^i \log \mathbf{y}_r^i \qquad (7)$$

where N denotes the number of training data, $\mathcal{R}$ denotes the number of relations, $\mathbf{p}_i$ is a one-hot vector, denotes the golden label of the i^{th} instance, $\mathbf{y}$ indicates the predicted distribution, r denotes the i^{th} value of the vector. We adopt AdamW[13] as the optimizer to train our model.

4 Experiments

In order to verify the effectiveness of our proposed method, we designed a extensive experiments.

4.1 Datasets

We conduct our experiments on the public dataset SemEval 2010 Task 8 [14]. The SemEval dataset contains a total of 10217 instances, including 8000 training instances and 2717 test instances. Among them, two entities of each instance have been given. Therefore, the task of relation extraction becomes to determine the semantic relation between two entities in the sentence. The SemEval dataset contains a total of 19 types of relation, including 9 pairs of directed relations and a special semantic relation "None", which means that there is no semantic relation between the two entities. In addition, SemEval officially provides evaluation metric.

4.2 Experimental Setup

For each instance, we use the existing natural language processing toolkit Stanford Corenlp Toolkit [1] as dependency parser, and we get a total of 40 dependencies. For encoder, we adopt bert-base-uncased version which has 12 layers and 768 hidden size. For the location of the entity in the sentence, we establish a mask to represent, where the position of 0 in the mask represents the non-entity location, and 1 represents the location of the entity. In order to obtain the most effective dependency type, we first set the corresponding value in $\mathcal{W}$ to 0.5. The settings of other hyperparameters are shown in Table 1

[1] https://github.com/stanfordnlp/CoreNLP

Hyperparameters	Value
weight decay	0.01
batch size	16
epochs	20
learning rate	1e-4

Table 1: Hyperparameters of our model

4.3 Baseline Methods

We compare the proposed method with the following baselines:

- GCN[15] proposes three pruning strategies. One is pruning the tree down to the dependency path, the other is keeping all nodes that are directly attached to the dependency path, and the last is retaining the entire lowest common ancestor (LCA) subtree.
- AGGCN[16] adopts the multi-head attention mechanism when constructing the adjacency matrix. In addition, in order to capture the semantic information of different granularity, each convolution layer also contains several sub-layers.
- A-GCN [17] pays attention to the type of dependency. When calculating the adjacency matrix $\mathbf{A}$, the generalized attention mechanism is adopted. Each entry in $\mathbf{A}$ is determined by the dependency type and the context representation of tokens. In the calculation process, a variety of different pruning strategies are adopted.

4.4 Main Results

We train our model for two times. In the first time, we take all the 40 dependency types into account. It is intuitively believed that the greater the change, the more sensitive the dependency type is to the model. Then we statistics the absolute value change of each value in $\mathcal{W}$ as shown in Figure 2. Finally, we select the Top-18 dependency type with the highest changes and conduct

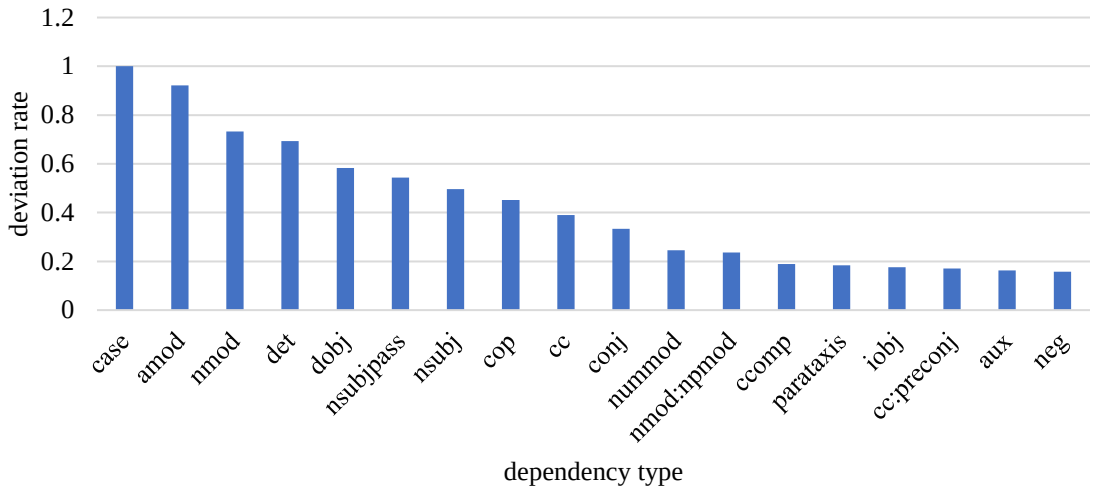

Figure 2: The overview of our model. (a) is an example sentence and its dependency parsing. (b) is the framework of our model. (c) is the adjacency matrix composed of the real values corresponding to the dependency type. The yellow and green parts in the sentence indicate the two entities in the sentence, respectively

	Prec.	**Rec.**	**F1-score**
GCN[15]	-	-	84.8
AGGCN[16]	-	-	85.7
A-GCN[17]	-	-	**89.16**
Ours(F)	87.94	88.29	88.05
Ours(S)	88.61	88.77	88.62

Table 2: Results on SemEval. Ours(F) denotes the first training result and Ours(S) denotes the fine-tuned result.

the experiment again to fine-tune the model. All experimental results are shown in Table 2.

From the table, we can get the following information: (1) From the first (F) and second (S) experimental results, we can conclude that the selection of dependency and the fine-tuning of the model have had an impact on the test results of the model, with a 0.57% improvement in F1-score. (2) Compare with other methods, GCN[15] and AGGCN[16], our model exceeds 3.82% and 2.92% in F1-score, respectively. (3) For A-GCN[17], our model also achieves comparable performance, compared with our model, A-GCN[17] has many additional modules and calculation costs.

5 Conclusion

In this paper, we propose a dependency selection method to prune the dependency tree. Our model first statistics the impact of each dependency type on the relation extraction task, and then we select the dependency types that are sensitive to the results to retain. The experiment shows that our method is effective.

References

[1] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. Learning intents behind interactions with knowledge graph for recommendation. In *Proceedings of the Web Conference 2021*, pages 878–887, 2021.

[2] Sasha Sheng, Amanpreet Singh, Vedanuj Goswami, Jose Magana, Tristan Thrush, Wojciech Galuba, Devi Parikh, and Douwe Kiela. Human-adversarial visual question answering. *Advances in Neural Information Processing Systems*, 34:20346–20359, 2021.

[3] Marouane Birjali, Mohammed Kasri, and Abderrahim Beni-Hssane. A comprehensive survey on sentiment analy-

sis: Approaches, challenges and trends. *Knowledge-Based Systems*, 226:107134, 2021.

[4] Amir Biglari and J. Sutherland. An a-posteriori evaluation of principal component analysis-based models for turbulent combustion simulations. *Combustion and Flame*, 162:4025–4035, 2015.

[5] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. Learning Intents behind Interactions with Knowledge Graph for Recommendation. In Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia, editors, *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 878–887. ACM / IW3C2, 2021.

[6] Zhenhailong Wang and Heng Ji. Open Vocabulary Electroencephalography-to-Text Decoding and Zero-Shot Sentiment Classification. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 5350–5358. AAAI Press, 2022.

[7] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1785–1794, 2015.

[8] Yuhao Zhang, Peng Qi, and Christopher D Manning. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215, 2018.

[9] Yuanhe Tian, Guimin Chen, Yan Song, and Xiang Wan. Dependency-driven relation extraction with attentive graph convolutional networks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4458–4471, 2021.

[10] Zhijiang Guo, Yan Zhang, and Wei Lu. Attention guided graph convolutional networks for relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 241–251, 2019.

[11] Bowen Yu, Xue Mengge, Zhenyu Zhang, Tingwen Liu, Wang Yubin, and Bin Wang. Learning to prune dependency trees with rethinking for neural relation extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3842–3852, 2020.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[13] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[14] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In Katrin Erk and Carlo Strapparava, editors, *Proceedings of*

the 5th International Workshop on Semantic Evaluation, SemEval@ACL 2010, Uppsala University, Uppsala, Sweden, July 15-16, 2010, pages 33–38. The Association for Computer Linguistics, 2010.

[15] Yuhao Zhang, Peng Qi, and Christopher D. Manning. Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2205–2215. Association for Computational Linguistics, 2018.

[16] Zhijiang Guo, Yan Zhang, and Wei Lu. Attention Guided Graph Convolutional Networks for Relation Extraction. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 241–251. Association for Computational Linguistics, 2019.

[17] Yuanhe Tian, Guimin Chen, Yan Song, and Xiang Wan. Dependency-driven Relation Extraction with Attentive Graph Convolutional Networks. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4458–4471. Association for Computational Linguistics, 2021.