Dependency Position Encoding for Relation Extraction

Qiushi Guo, Xin Wang, Dehong Gao*
Alibaba Group
{qiushi.gqs,lucas.wangx,dehong.gdh}@alibaba-inc.com

Abstract

Leveraging the dependency tree of the input sentence is able to improve the model performance for relation extraction. A challenging issue is how to remove confusions from the tree. Efforts have been made to utilize the dependency connections between words to selectively emphasize target-relevant information. However, these approaches are limited in focusing on exploiting dependency types. In this paper, we propose dependency position encoding (DPE), an efficient way of incorporating both dependency connections and dependency types into the self-attention mechanism to distinguish the importance of different word dependencies for the task. In contrast to previous studies that process input sentence and dependency information in separate streams, DPE can be seamlessly incorporated into the Transformer and makes it possible to use an one-stream scheme to extract relations between entity pairs. Extensive experiments show that models with our DPE significantly outperform the previous methods on SemEval 2010 Task 8, KBP37, and TACRED.

1 Introduction

Relation extraction (RE) has been a long standing goal in natural language processing (NLP) and plays a crucial role in supporting many downstream task (Trisedya et al., 2019; Sun et al., 2019; Xu et al., 2016; Wang and Cardie, 2012). Nowadays, methods with powerful encoders (*e.g.* Transformer) have achieved promising success in RE (Baldini Soares et al., 2019; Tian et al., 2021; Yu et al., 2020; Guo et al., 2019; Mandya et al., 2020) due to their effectiveness in capturing contextual information. In addition, previous studies (Miwa and Bansal, 2016; Zhang et al., 2018; Sun et al., 2020; Chen et al., 2021) try to utilize the extra syntactic knowledge (*e.g.* word dependency) to further improve the ability to encode relations between entity pairs and have demonstrated its benefit in many methods. Nevertheless, intensively leveraging dependency information is not able to always improve the model performance, due to the confusions introduced by the noise in the dependency tree. Efforts have been made to utilize the separate module to selectively emphasize target-relevant dependency connections between words, with little attention paid to dependency types. We argue that dependency types associated with the dependency connections are able to help the relation extraction task and try to find a direct way to infuse them into the model.

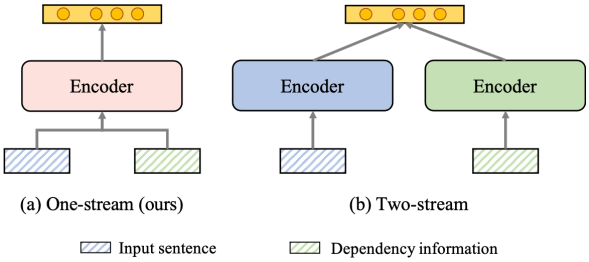

Figure 1: We introduce an one-stream scheme for relation extraction. By modeling the dependency information as a labeled and fully-connected graph, we extend self-attention to consider words dependencies and seamlessly incorporate them into Transformer-based encoder.

In this paper, we present dependency position encoding (DPE), an efficient way of incorporating dependency information into the self-attention mechanism of the Transformer to distinguish the importance of different word dependencies for relation extraction. Specially, we first encode dependency information obtained from an off-the-shelf dependency parser and map it into embeddings, then assign different weights to different labeled dependency connections between any two words through attention calculation. Different from previous studies that process input sentence and dependency tree in separate streams, our DPE can be seamlessly incorporated into the encoder and makes it possible

*Corresponding author.

Findings of the Association for Computational Linguistics: NAACL 2022, pages 1601 - 1606

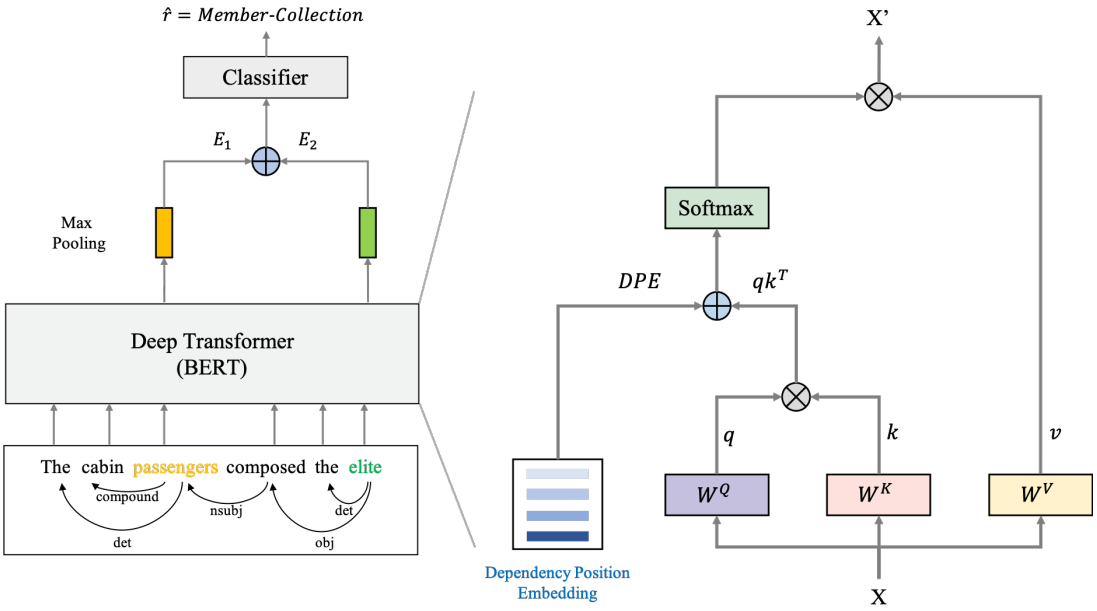

Figure 2: Relation extraction with dependency position embedding. The left side shows the overall architecture, while on the right side, we illustrate the detail of extending the self-attention to consider word dependencies.

to introduce an one-stream scheme for relation extraction. We conduct experiments on three English benchmark datasets (i.e., SemEval 2010 Task 8, KBP37 and TACRED) where the results demonstrate the effectiveness of our method.

2 Background and Motivation

BERT (Devlin et al., 2019) is a pre-trained language model which has Transformer-based model architecture and is widely used as sentence encoder in RE methods (Baldini Soares et al., 2019; Wu and He, 2019; Tian et al., 2021). The self-attention mechanism in Transformer does not explicitly model relative or absolute position information. To this end, Transformer (Vaswani et al., 2017) adopts explicit absolute sinusoidal positional encodings added into input embeddings. Absolution positions can be more naturally encoded as relative positional encodings (RPE) as follows:

$$
\begin{aligned}
Attn(x_i) &= \sum_{j=1}^{n} \alpha_{ij}(W^V x_j + a_{ij}^V) \\
\alpha_{ij} &= \frac{exp\ e_{ij}}{\sum_{k=1}^{n} exp\ e_{ik}} \\
e_{ij} &= \frac{x_i W^Q (x_j W^K + a_{ij}^K)^T}{\sqrt{d_k}}
\end{aligned} \tag{1}
$$

where $a_{ij} \in \mathbb{R}^{d_k}$ denotes edge distances between x_i and x_j when seeing input elements as labeled, directed, fully-connected graph (Shaw et al., 2018). Inspired by RPE, we argue that dependency types associated with the dependency connections among words also can be regarded as dependency graphs, which contain highly useful information for RE. We propose dependency positional encoding (DPE) and use it as an extension of self-attention to consider the dependency information between words. Different from previous studies (Guo et al., 2019; Zhang et al., 2018) that treat the dependency connections among words equally, DPE is able to distinguish the importance of different labeled connections through self-attention calculation.

3 Methodology

Following the previous studies, we perform RE as a classification task. Specifically, given an input sentence $x = (x_1, ..., x_n)$ with $E1$ and $E2$ denoting two entities in it, we predict the relation $\hat{r}$ between entities as fowllows:

$$
\hat{r} = \underset{r \epsilon R}{arg\,max}\ p(r|DPE(x, \tau_x)) \tag{2}
$$

where R is the set of entity relation types and τ_x is the dependency tree obtained from an off-the-shelf toolkit. The overall architecture of our method is illustrated in Figure 2 and we impose the dependency information into the model through DPE.

3.1 Dependency Position Encoding

Dependency tree τ_x indicates type-aware connections between input elements. We firstly repre-

MODEL	F_1-score
SDP-LSTM (Xu et al., 2015)	83.7
PA-LSTM (Zhang et al., 2017)	82.7
Att-Pooling-CNN (Wang et al., 2016)	88.0
C-GCN (Zhang et al., 2018)	84.8
R-BERT (Wu and He, 2019)	89.2
LST-AGCN (Sun et al., 2020)	86.0
SPTree (Miwa and Bansal, 2016)	84.4
C-AGGCN (Guo et al., 2019)	85.7
C-GCN-MG (Mandya et al., 2020)	85.9
DP-GCN (Yu et al., 2020)	86.4
$BERT_{EM}$+MTB (Baldini Soares et al., 2019)	89.5
A-GCN (Tian et al., 2021)	89.8
TaMM (Chen et al., 2021)	90.0
DPE (BERT-base)	89.2
DPE (BERT-large)	**90.2**

Table 1: The comparison between our models and previous studies on SemEval.

sent dependency types in τ_x by a type matrix $T = (t_{i,j})_{n \times n}$, where $t_{i,j}$ is the dependency type (*e.g.* nusbj) associated with the directed dependency connection between x_i and x_j. We then map each $t_{i,j}$ to its embedding $e_{i,j}$. For each Transformer layer, we operate self-attention calculation as follows:

$$Att(x) = Softmax(\frac{QK^T + DPE}{\sqrt{d_k}}) \cdot V$$
$$Q = W^Q x, K = W^K x, V = W^V x \quad (3)$$

where W^Q, W^K, and W^V are weight matrices to generate Q, K, and V via linear transformations on x; DPE denotes the type embedding $e_{i,j}$. DPE can be seamlessly incorporated to the Transformer framework and and infuse word dependencies into the model in an one-stream manner.

3.2 Relation Extraction with DPE

We first encode the input sentence into hidden vectors by BERT (Devlin et al., 2019) with DPE, where h_i denotes the hidden vector for x_i. Then, we apply the max pooling to the output hidden vector of each word in the entity to obtain the entity representation h_{E_k} by

$$h_{E_k} = MaxPooling(h_i | x_i \epsilon E_k) \quad (4)$$

Next, we concatenate the representation of the two entities and pass the resulting vector through a fully connected layer to obtain the final prediction $\hat{r}$, as follows:

$$\hat{r} = W \cdot (h_{E_1} + h_{E_2}) + b \quad (5)$$

MODEL	F_1-score
RNN (Zhang and Wang, 2015)	58.8
$BERT_{EM}$ (Baldini Soares et al., 2019)	68.3
DPE (BERT-base)	67.1
DPE (BERT-large)	**68.7**

Table 2: Performance of different models on KBP37.

MODEL	P	R	F_1
SDP-LSTM (Xu et al., 2015)	66.3	52.7	58.7
Tree-LSTM (Tai et al., 2015)	66.0	59.2	62.4
C-GCN-MG (Mandya et al., 2020)	68.0	64.4	66.1
AGGCN (Guo et al., 2019)	69.9	60.9	65.1
C-AGGCN (Guo et al., 2019)	71.8	66.4	69.0
$BERT_{EM}$ (Baldini Soares et al., 2019)	-	-	70.1
GCN (Zhang et al., 2018)	69.8	59.0	64.0
C-GCN (Zhang et al., 2018)	69.9	63.3	66.4
PA-LSTM (Zhang et al., 2017)	65.7	64.5	65.1
DP-GCN (Yu et al., 2020)	72.2	66.5	69.2
DPE (BERT-base)	68.2	64.0	66.0
DPE (BERT-large)	**75.0**	63.1	68.5

Table 3: Results of different models on TACRED.

where W and b are the trainable weight matrix and bias vector for the fully connected layer.

4 Experiments

4.1 Experimental Settings

Datasets. We conduct experiments of relation extraction on three English benchmark datasets: SemEval 2010 Task 8 (SemEval)* (Hendrickx et al., 2019), KBP37 (Zhang and Wang, 2015) and TACRED† (Zhang et al., 2017). For fair comparison, we use their official train/dev/test split‡. Following previous studies (Hendrickx et al., 2019; Wang et al., 2016; Baldini Soares et al., 2019; Yu et al., 2020), we report the macro-averaged F_1 scores on SemEval and the micro-averaged F_1 scores on KBP37 and TACRED.

Dependency Information Construction. We employ Standard CoreNLP Toolkits (SCT) § to obtain the dependency tree τ_x for each input sentence x. Motivated by previous studies (Xu et al., 2015; Zhang et al., 2018), we use two groups of dependency connections which are filtered out through particular pruning strategies: (1) *full connections*

*The data is download from http://docs.google.com/View?docid=dfvxd49s_36c28v9pmw.

†We obtain the official data (LDC2018T24) from https://catalog.ldc.upenn.edu/LDC2018T24.

‡SemEval only has the training and test sets.

§We download the version 3.9.2 from https://stanfordnlp.github.io/CoreNLP/.

Model	Order	SemEval	KBP37	TACRED
Baseline	-	89.2	67.5	67.4
DPE (Full)	1st	89.6	67.8	67.7
	2nd	89.8	67.9	67.8
	3rd	89.9	68.3	68.3
DPE (Part)	1st	89.9	68.5	68.4
DPE (Both)	1st	**90.2**	**68.7**	**68.5**
	2nd	89.9	68.4	68.2
	3rd	89.6	67.9	67.8

Table 4: Ablation study of different dependency information.

Model	Position	SemEval	KBP37	TACRED
Baseline	-	89.2	67.5	67.4
DPE (Full)	1-12	89.4	67.7	67.5
	13-24	89.5	67.6	67.6
	1-24	**89.6**	**67.8**	**67.7**
DPE (Part)	1-12	89.8	68.2	68.1
	13-24	89.7	68.3	68.2
	1-24	**89.9**	**68.5**	**68.4**
DPE (Both)	1-12	89.6	68.5	68.2
	13-24	89.8	68.3	68.3
	1-24	**90.2**	**68.7**	**68.5**

Table 5: Ablation study of different plugin positions.

include all dependencies that directly connect to the heads of two entities; (2) *part connections* are obtained from the shortest dependency path (SDP) between entities.

Implementation Details. We use the uncased version of BERT (Devlin et al., 2019) (we utilize 24 layers of multi-head attentions with 1024-dimensional hidden vectors for BERT-large) with DPE and insert four special tokens (*i.e.*, "<*e1*>", "</*e1*>", "<*e2*>", and "</*e2*>") into the input sentence to mark the boundary of the two entities (Baldini Soares et al., 2019). We randomly initialize all trainable parameters and the dependency type embeddings. We utilize Adam optimizer with the initial learning rate of 3e-5 and the batch size of 64 to train the model.

4.2 Main Results

In table 1, we compare the the F_1 scores of different models on the test set of SemEval (Hendrickx et al., 2019). As can be seen, our model consistently outperforms other methods and achieves the best F_1 score. To further demonstrate the advantage of DPE, we also conduct experiments on the KBP37 and TACRED datasets. Table 2 and Table 3 show that our method also achieves strong performance on these two benchmark datasets. Above results illustrate the great generalizability of DPE in condensing the useful dependency information for relation extraction.

4.3 Analysis

The Effect of Dependency Information. We study the performance of our proposed model with different combinations and different orders of word dependencies. As we can see in Table 4, models with DPE under all settings outperform the BERT-large baseline on all datasets. We find that DPE models with part connections (*i.e.*, DPE (Part)) surpass the ones with full connections (*i.e.*, DPE (Full)) under the same setting since using the dependency information in an intensive way may introduce noise. Besides, we obtain the best performance when using two types of word dependencies (*i.e.*, DPE (Both)). Furthermore, we observe that DPE (Full) models obtain better results when using higher order dependencies while the trend is on the opposite for the DPE (Both) ones. One possible reason is that leveraging higher order dependencies makes it possible to capture more useful contextual information between two entities for DPE (Full) and introduces confusions when DPE (Both) encodes most essential word dependencies.

The Plugin Positions of DPE. We also experiment by varying the position of the DPE in the model. Table 5 presents the ablations for variable positions based on the BERT-large. For fair comparison, we use the model on first-order word dependencies used in the above analysis. We denote the first BERT layer by index 1 and $i - j$ as the plugin positions of DPE which start from i and end at j. We achieve the best performance when each layer in the BERT-large equipped with DPE across all combinations of dependency information, which demonstrates the effectiveness of our method.

5 Conclusion

In this paper, we propose dependency position encoding (DPE) and use it as an extension of self-attention to consider dependency information. Different from previous studies, DPE is able to seamlessly incorporated into the Transformer and makes it possible to use an one-stream scheme to extract illuminating word dependencies for relation extraction. Experimental results on several public datasets illustrate the effectiveness of our approach.

References

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Guimin Chen, Yuanhe Tian, Yan Song, and Xiang Wan. 2021. Relation extraction with type-aware map memories of word dependencies. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2501–2512, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid O Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2019. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. *arXiv preprint arXiv:1911.10422*.

Angrosh Mandya, Danushka Bollegala, and Frans Coenen. 2020. Graph convolution over multiple dependency sub-graphs for relation extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6424–6435, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Kai Sun, Richong Zhang, Yongyi Mao, Samuel Mensah, and Xudong Liu. 2020. Relation extraction with convolutional network over learnable syntax-transport graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8928–8935.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5679–5688, Hong Kong, China. Association for Computational Linguistics.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China. Association for Computational Linguistics.

Yuanhe Tian, Guimin Chen, Yan Song, and Xiang Wan. 2021. 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, Online. Association for Computational Linguistics.

Bayu Distiawan Trisedya, Gerhard Weikum, Jianzhong Qi, and Rui Zhang. 2019. Neural relation extraction for knowledge base enrichment. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 229–240, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. 2016. Relation classification via multi-level attention CNNs. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1298–1307, Berlin, Germany. Association for Computational Linguistics.

Lu Wang and Claire Cardie. 2012. Focused meeting summarization via unsupervised relation extraction. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 304–313, Seoul, South Korea. Association for Computational Linguistics.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for

relation classification. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2361–2364.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question answering on Freebase via relation extraction and textual evidence. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2326–2336, Berlin, Germany. Association for Computational Linguistics.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. 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, Lisbon, Portugal. Association for Computational Linguistics.

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

Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. 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, Brussels, Belgium. Association for Computational Linguistics.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45, Copenhagen, Denmark. Association for Computational Linguistics.