Relation Extraction with Word Graphs from N-grams

Han Qin[♠*], **Yuanhe Tian**[♥*], **Yan Song**[♠†]
[♠]The Chinese University of Hong Kong (Shenzhen) [♥]University of Washington
[♠]hanqin@link.cuhk.edu.cn
[♥]yhtian@uw.edu [♠]songyan@cuhk.edu.cn

Abstract

Most recent studies for relation extraction (RE) leverage the dependency tree of the input sentence to incorporate syntax-driven contextual information to improve model performance, with little attention paid to the limitation where high-quality dependency parsers in most cases unavailable, especially for in-domain scenarios. To address this limitation, in this paper, we propose attentive graph convolutional networks (A-GCN) to improve neural RE methods with an unsupervised manner to build the context graph, without relying on the existence of a dependency parser. Specifically, we construct the graph from n-grams extracted from a lexicon built from pointwise mutual information (PMI) and apply attention over the graph. Therefore, different word pairs from the contexts within and across n-grams are weighted in the model and facilitate RE accordingly. Experimental results with further analyses on two English benchmark datasets for RE demonstrate the effectiveness of our approach, where state-of-the-art performance is observed on both datasets.[1]

1 Introduction

Recently, neural models (Zeng et al., 2014; Zhang and Wang, 2015; Xu et al., 2015; dos Santos et al., 2015; Zhang et al., 2015; Wang et al., 2016; Zhou et al., 2016; Zhang et al., 2017; Soares et al., 2019) with powerful encoders (e.g., Transformers) have achieved promising performance for relation extraction (RE), for the reason that the encoders are superior in capturing contextual information and thus allow RE systems to better understand the text and correctly identify the relations between entities in the given text. To further improve the ability of RE models to understand the context, many studies (Xu et al., 2016; Zhang et al., 2018; Guo et al., 2019; Sun et al., 2020; Yu et al., 2020; Mandya et al., 2020; Tian et al., 2021d; Chen et al., 2021) leverage extra resources, such as auto-parsed word dependency, through graph-based approaches, e.g., graph convolutional networks (GCN). In doing so, such studies learn the long-distance connections among useful words from the dependency tree and extract relations between entity pairs accordingly. However, in doing so, dependency parsers required by their approaches are not always available. In this dilemma, one needs another way to extract useful word connections to build the graph for GCN-based models, whereas limited attentions from previous studies have been paid to this alternative.

In this paper, we propose attentive GCN (A-GCN) for relation extraction, where its input graph is built based on n-grams extracted with unsupervised methods i.e., pointwise mutual information (PMI), rather than an existing dependency parser. Specifically, two types of edges in the graph are introduced to model word connections within and across n-grams and an attention mechanism is applied to GCN to weight these edges. In doing so, different contextual information are discriminatively learned to facilitate RE without requiring any external resources. We evaluate our approach on two English benchmark datasets, i.e., ACE2005EN and SemEval 2010 Task 8, where the results demonstrate the effectiveness of our approach with state-of-the-art performance observed on both datasets.

2 The Approach

RE is often treated as a classification task, where the input is a sentence $\mathcal{X} = x_1, \cdots, x_n$ with two given entities (denoted by E_1 and E_2) in it. Our approach follows this paradigm and uses a variant of graph neural model, i.e., attentive GCN (A-GCN), to incorporate word pair information and predicts the relation $\widehat{r}$ between E_1 and E_2 by

$$\widehat{r} = \arg\max_{r \in \mathcal{R}} \; p\left(r | \text{A-GCN}\left(\mathcal{X}, \mathcal{G}_{\mathcal{X}}, E_1, E_2\right)\right) \quad (1)$$

[*]Equal contribution.
[†]Corresponding author.
[1]The code involved in this paper are released at https://github.com/cuhksz-nlp/RE-NGCN.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 2860–2868
November 7–11, 2021.

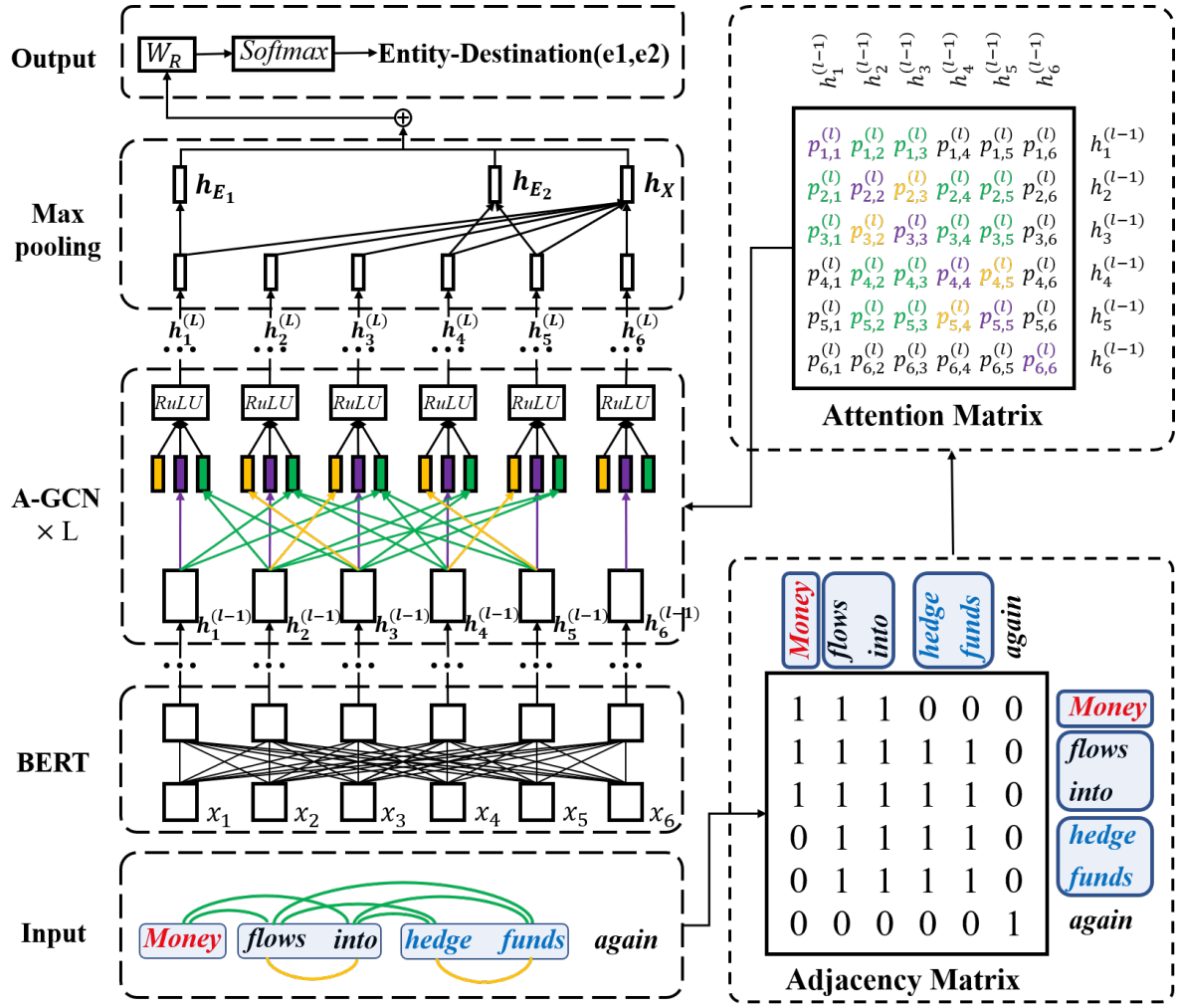

Figure 1: An overview of the architecture for A-GCN with the graph built upon n-grams illustrated in blue boxes. Two given entities (i.e., "*Money*" and "*hedge funds*") are shown in red and blue colors, respectively.

where $\mathcal{G}_{\mathcal{X}}$ is the graph built based on n-grams in $\mathcal{X}$, $\mathcal{R}$ is the relation type set; p computes the probability of a particular relation type $r \in \mathcal{R}$ with the given input (i.e., $\mathcal{X}$, $\mathcal{G}_{\mathcal{X}}$, E_1, and E_2), and $\widehat{r}$ is the prediction of our A-GCN model. In the following text, we firstly elaborate how we construct the graph based on n-grams, and then illustrate the architecture of the A-GCN model for RE.

2.1 Graph Construction from N-grams

Conventionally, the graph used in GCN-based models for natural language understanding tasks (including RE) is constructed by the dependency tree of each input sentence. However, high-quality dependency parsers are not always available. Therefore, we do not want our model to rely on the existence of dependency parsers and hence we need an alternative to build the graph. Given that n-grams are widely used as effective features that carry contextual information to enhance the model performance in many previous studies (Song et al., 2009; Song and Xia, 2012; Ishiwatari et al., 2017; Yoon et al., 2018; Tian et al., 2020a,b,c, 2021a), we propose to construct the graph for GCN-based models based on n-grams in $\mathcal{X}$ which are extracted from a pre-constructed n-gram lexicon $\mathcal{N}$.

N-gram Lexicon Construction Before we segment appropriate n-grams for each input sentence, an n-gram lexicon $\mathcal{N}$ is built over the entire corpus based on pointwise mutual information (PMI). Specifically, we firstly compute the PMI of any two adjacent words x', x'' for all data by

$$PMI(x', x'') = \log \frac{p(x'x'')}{p(x')p(x'')} \quad (2)$$

where p is the probability of an n-gram (i.e., x', x'' and $x'x''$) in the training set; thus a higher PMI score suggests a greater chance of forming an n-gram. Therefore, for each pair of adjacent words x_{i-1}, x_i, we use a predefined threshold to determine whether the two words should be combined or split. Through this process, we segment all sen-

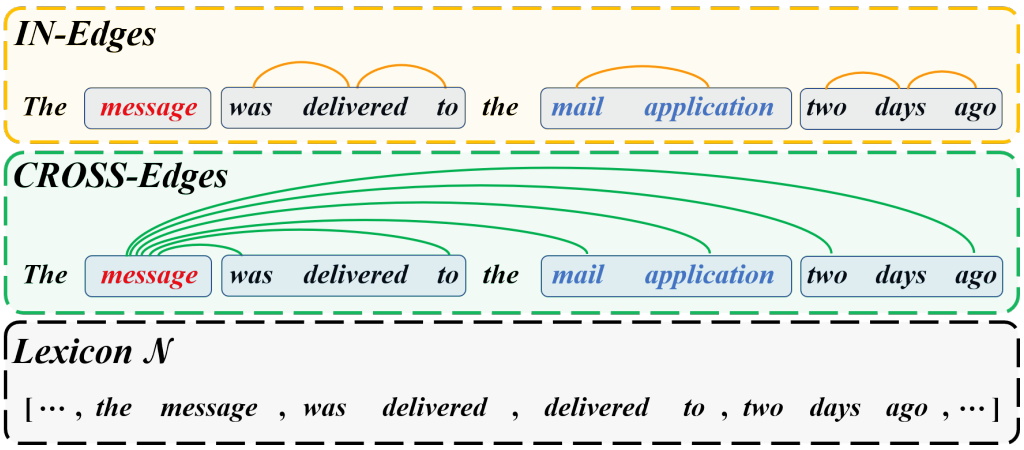

Figure 2: Examples of *IN* and *CROSS* edges for building the graph in an example input sentence. Herein, n-grams extracted from the lexicon $\mathcal{N}$ are shown in the bottom gray box; the two entities (i.e., "*message*" and "*mail application*") are highlighted in red and blue colors; example *IN* and *CROSS* edges are marked in yellow and green colors, respectively (for simplicity, we only show *CROSS* edges associated with "*message*").

tences in the training set into small text spans and collect them to construct the n-gram lexicon $\mathcal{N}$.

N-gram Extraction for Each Sentence Based on the given entities (i.e., E_1 and E_2) and the n-gram lexicon $\mathcal{N}$, n-grams in a sentence are extracted as follows. First, each entity itself is considered to be an n-gram. Then, we extracts n-grams appearing in $\mathcal{N}$ from the sentence, where if there are overlaps between n-grams, we merge them into a larger n-gram. For example, we extract four n-grams (i.e., "*message*", "*was delivered to*", "*mail application*", and "*two days ago*" illustrated in blue boxes) from the example sentence shown in Figure 2. In these n-grams, "*two days ago*" is a non-overlapping n-gram included in the lexicon; "*was delivered to*" is the merger of two overlapping n-grams "*was delivered*" and "*delivered to*"; "*message*" and "*mail application*", highlighted in red and blue respectively, are the n-grams for given entities. In general, word-word connections of adjacent words in the same n-gram is strong in terms of co-occurrence, as well as some connections from words among co-occurred n-grams, which motivates us to treat such connections as important edges in the graph for GCN-based models.

Graph Construction Given an input sentence $\mathcal{X}$ with extracted n-grams, we construct the graph for GCN-based model via two types of undirected edges, i.e., the "*IN*" and "*CROSS*" edges. The first type is to model the local word pairs while the second type is to model the word pairs cross n-grams. For the first type, any two adjacent words within the same n-gram are connected. For the second type, inspired by that English phrases tend to be either head-initial or head-final (e.g., phrase "*read some books*" and "*green apples*" respectively) in many cases, we connect the starting and ending words of any two n-grams under the condition that to no more than two n-grams between them. As an illustration, Figure 2 shows all *IN* edges (highlighted in yellow) and some *CROSS* edges (highlighted in green) for an example sentence.[2]

2.2 Attentive Graph Convolutional Networks

Standard GCN models treat all word pairs from the graph equally and hence are not able to handle the possibility that different x_i may contribute separately to x_j. Especially for n-gram based graph construction, it is of vital significance that the proposed model distinguish different word pairs since all n-grams and the graph are constructed automatically without any supervised guidance. Therefore, we apply an attention mechanism to the adjacent matrix $\mathbf{A} = (a_{i,j})_{n\times n}$ of the graph ($a_{i,j} = 1$ if $i = j$ or there is an *IN* or *CROSS* edge between x_i and x_j in the graph; $a_{i,j} = 0$, otherwise), where a weight $p_{i,j}^{(l)}$ is attached to each x_i and its associated x_j in the l-th A-GCN layer. Formally, this process to compute $p_{i,j}^{(l)}$ can be presented by

$$p_{i,j}^{(l)} = \frac{a_{i,j} \cdot \exp\left(\mathbf{h}_i^{(l-1)} \cdot \mathbf{h}_j^{(l-1)}\right)}{\sum_{j=1}^{n} a_{i,j} \cdot \exp\left(\mathbf{h}_i^{(l-1)} \cdot \mathbf{h}_j^{(l-1)}\right)} \quad (3)$$

where $h_i^{(l-1)}$ refers to the output vector for x_i from the previous GCN layer and "$\cdot$" denotes the inner

[2]We only show the *CROSS* edges associated with "*message*" for simplicity. There are other *CROSS* edges (e.g., the edges between word pairs ("*was*", "*two*"), ("*was*","*ago*"), and ("*to*", "*two*"), etc.) that are not shown in the figure.

production. Afterwards, we apply weight $p_{i,j}^{(l)}$ to the connection between x_i and x_j and obtain the output representation $h_i^{(l)}$ by

$$\mathbf{h}_i^{(l)} = \sigma\left(\sum_{j=1}^{n} p_{i,j}^{(l)}\left(\mathbf{W}_{u_{i,j}}^{(l)} \cdot \mathbf{h}_j^{(l-1)} + \mathbf{b}_{u_{i,j}}^{(l)}\right)\right) \quad (4)$$

where σ denotes the $ReLU$ activation function. In addition, $\mathbf{W}_{u_{i,j}}^{(l)}$ and $\mathbf{b}_{u_{i,j}}^{(l)}$ have three choices (i.e., $\mathbf{W}_{\text{in}}^{(l)}$, $\mathbf{W}_{\text{cross}}^{(l)}$, and $\mathbf{W}_{\text{self}}^{(l)}$ for $\mathbf{W}_{u_{i,j}}^{(l)}$ and $\mathbf{b}_{\text{in}}^{(l)}$, $\mathbf{b}_{\text{cross}}^{(l)}$, and $\mathbf{b}_{\text{self}}^{(l)}$ for $\mathbf{b}_{u_{i,j}}^{(l)}$) depending on the edge type (i.e., *IN*, *CROSS*, and self-connected edges) between x_i and x_j.[3] Compared with standard GCN, our approach is able to attach different numerical weights to word pairs and distinguish the importance of them so as to better leverage contextual information accordingly. Moreover, we integrate the edge type information into the output representation of x_i (i.e., $\mathbf{h}_i^{(l)}$), so that different types of contextual information are separately modeled.

2.3 Relation Extraction with A-GCN

To conduct relation extraction with A-GCN, we obtain the hidden vector $\mathbf{h}_i^{(0)}$ for x_i from BERT (Devlin et al., 2019) to feed into the first A-GCN layer. Next, we apply the forward function (i.e., Eq. (3)-(4)) in each A-GCN layer and obtain the output (i.e., $\mathbf{h}_i^L$) from the last A-GCN layer (i.e. the L-th layer). Then, we apply max pooling to all words as well as the words belong to an entity so as to obtain the representation of the entire sentence $\mathbf{h}_{\mathcal{X}}$ and the two entities $\mathbf{h}_{E_k}$ (i.e. $k = 1, 2$), respectively. This process is thus formalized by

$$\mathbf{h}_{\mathcal{X}} = \text{MaxPooling}(\{\mathbf{h}_1^{(L)}, \cdots, \mathbf{h}_n^{(L)}\}) \quad (5)$$

and

$$\mathbf{h}_{E_k} = \text{MaxPooling}(\{\mathbf{h}_i^{(L)} | x_i \in E_k\}) \quad (6)$$

Afterwards, we concatenate the representations of the sentence (i.e., $\mathbf{h}_{\mathcal{X}}$) and the two entities (i.e., $\mathbf{h}_{E_1}$ and $\mathbf{h}_{E_2}$) and apply a trainable matrix $\mathbf{W}_R$ to the resulting vector to map it to the output space by

$$\mathbf{o} = \mathbf{W}_R \cdot (\mathbf{h}_{\mathcal{X}} \oplus \mathbf{h}_{E_1} \oplus \mathbf{h}_{E_2}) \quad (7)$$

where $\mathbf{o}$ is a $|\mathcal{R}|$-dimensional vector with each of its value referring to a relation type from the label set $\mathcal{R}$. Finally, we apply a *softmax* function to $\mathbf{o}$ to predict the relation $\widehat{r}$ between E_1 and E_2.

[3]For example, if the edge type between x_i and x_j is *CROSS*, then $\mathbf{W}_{u_{i,j}}^{(l)} = \mathbf{W}_{\text{cross}}^{(l)}$ and $\mathbf{b}_{u_{i,j}}^{(l)} = \mathbf{b}_{\text{cross}}^{(l)}$.

3 Experiments

3.1 Settings

We run experiments on two English benchmark datasets for RE, namely, ACE2005EN (ACE05)[4] and SemEval 2010 Task 8 (SemEval)[5] (Hendrickx et al., 2010). For both datasets, we follow previous studies (Miwa and Bansal, 2016; Christopoulou et al., 2018; Ye et al., 2019) to pre-process them and split them into train/dev/test splits.[6] To build the n-gram lexicon for graph construction, we perform PMI on the training set to extract n-grams whose lengths are within [1,5], with the threshold of the PMI score set to 0. For textual encoder, since the high-quality text representations are proved to be effective in enhancing the model performance (Mikolov et al., 2013; Song et al., 2018a,b; Song and Shi, 2018; Devlin et al., 2019; Yang et al., 2019; Song et al., 2021) and BERT is be able to provide high-quality text representations for natural language processing downstream tasks (Wu and He, 2019; Soares et al., 2019; Chen et al., 2020; Nie et al., 2020; Tian et al., 2021b,c; Qin et al., 2021a,b), we use BERT-large encoder[7] as our textual encoder. Moreover, we run standard GCN and our A-GCN models with two layers in the experiments. In addition to the proposed n-gram based graph construction, we also try fully connected graph (where every two words are connected through an edge) for both GCN and A-GCN. For evaluation, we follow previous studies to assess all models with F1 scores on the test sets.[8]

3.2 Results

Tabel 1 reports the F1 scores of different models on the test set of ACE05 and SemEval, where the results from BERT-large baseline (ID: 1) without using GCN are also reported for reference.[9] There are several observations. First, although the baseline (ID: 1) has achieved outstanding performance, our models with A-GCN (ID: 3, 5) still further im-

[4]We obtain the official data (LDC2006T06) from https://catalog.ldc.upenn.edu/LDC2006T06.

[5]The data is downloaded from http://docs.google.com/View?docid=dfvxd49s_36c28v9pmw.

[6]Detailed information for both datasets is in Appendix A.

[7]We download the uncased BERT-large from https://github.com/huggingface/transformers and use its default settings shown in Appendix B.

[8]We report the number of trainable parameters of different models as well as their inference speed in Appendix C.

[9]For the same group of models, we report the F1 scores on the development sets in Appendix D and the mean and standard deviation of their test set results in Appendix E.

ID	Models	ACE05	SemEval
1	BERT-large	76.12	89.00
2	+ GCN (Full)	76.94	89.12
3	+ A-GCN (Full)	77.43	89.21
4	+ GCN (N-gram)	77.29	89.17
5	+ A-GCN (N-gram)	**77.72**	**89.67**

Table 1: F1 scores of our A-GCN models and the baselines (i.e., BERT-only and standard GCN). "Full'' and "N-gram" represent the graph constructed based on all word connections and our approach, respectively.

prove the performance. This observation confirms the effectiveness of A-GCN and the graphs from n-grams. Second, both standard GCN and A-GCN with the graphs from n-grams (ID: 4,5) consistently outperform the ones with the fully connected graph (ID:2,3). Particularly, when the full graph is used, GCN obtains limited improvements over the baseline (ID: 1) or even worsen the performance, which are largely due to the noise introduced in the full graph. On the contrary, the graph built upon the n-grams only has the edges that connect important context words and thus allows GCN and A-GCN models to outperform the BERT-large baseline and achieve higher performance than the models with fully connected graph. Third, for both datasets, A-GCN functionalizes better than GCN (ID: 2, 3, 4, 5) with the same graph (i.e., either "Full" or"N-gram"). This observation is explained by that the attention mechanism distinguishes different edges of a graph by assigning higher weights to more important ones, so that facilitate relation extraction.

In addition, we compare our best models (i.e., BERT-large + A-GCN (N-gram)) with previous studies and report the results in Table 2, where our model outperforms all previous studies and achieves state-of-the-art performance on the two benchmark datasets. The result confirms that, compared with the dependency tree, graphs from n-grams also have a strong ability to extract contextual information. Moreover, although a graph from n-grams potentially carries some noise, the attention mechanism significantly helps to identify useful connections and facilitates RE accordingly, where different word pairs within and across n-grams are weighted and thus the model discriminatively learns from contextual information.

4 Conclusion

In this paper, we propose A-GCN to leverage information of word pairs for RE, where the graph for A-GCN is built from n-grams without relying on syntactic parsing. Particularly, we use PMI to extract n-grams from all training data and apply different connections among n-grams for graph networks, where attention is equipped to further enhance model performance. In doing so, A-GCN is able to dynamically learn from different word pairs so that less-informative relations are smartly pruned. Experimental results and analyses on two English benchmark datasets for RE demonstrate the effectiveness of our approach, where state-of-the-art performance is obtained on both datasets.

Models	ACE05	SemEval
Xu et al. (2015)	-	83.7
Wang et al. (2016)	-	88.0
Zhou et al. (2016)	-	84.0
Zhang et al. (2018)	-	84.8
Christopoulou et al. (2018)	64.2	-
Ye et al. (2019)	68.9	-
Wu and He (2019) (BERT)	-	89.2
Soares et al. (2019) (BERT)	-	89.5
Mandya et al. (2020)	-	85.9
Yu et al. (2020)	-	86.4
Wang et al. (2020)	66.7	-
Wang and Lu (2020)	67.6	-
Ours (BERT + A-GCN (N-gram))	**77.72**	**89.67**

Table 2: The comparison (F1 scores) between previous studies and our best models (i.e., BERT-large + A-GCN (N-gram)) on ACE05 and SemEval. Previous studies that leverage word dependencies are marked by "*".

Acknowledgements

This work is supported by NSFC under the project "The Essential Algorithms and Technologies for Standardized Analytics of Clinical Texts" (12026610) and Shenzhen Science and Technology Program under the project "Fundamental Algorithms of Natural Language Understanding for Chinese Medical Text Processing". It is also supported by Shenzhen Institute of Artificial Intelligence and Robotics for Society under the project "Automatic Knowledge Enhanced Natural Language Understanding and Its Applications" (AC01202101001).

References

Guimin Chen, Yuanhe Tian, and Yan Song. 2020. Joint Aspect Extraction and Sentiment Analysis with Directional Graph Convolutional Networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 272–279.

Guimin Chen, Yuanhe Tian, Yan Song, and Xiang Wan. 2021. Relation Extraction with Type-aware Map Memories of Word Dependencies. *Findings of the*

Association for Computational Linguistics: ACLIJCNLP.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2018. A Walk-based Model on Entity Graphs for Relation Extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 81–88.

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.

Cícero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying Relations by Ranking with Convolutional Neural 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 626–634.

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.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38.

Shonosuke Ishiwatari, Jingtao Yao, Shujie Liu, Mu Li, Ming Zhou, Naoki Yoshinaga, Masaru Kitsuregawa, and Weijia Jia. 2017. Chunk-based decoder for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1901–1912, Vancouver, Canada.

Angrosh Mandya, Danushka Bollegala, and Frans Coenen. 2020. Graph Convolution over Multiple Dependency Sub-graphs for Relation Extraction. In *COLING*, pages 6424–6435. International Committee on Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

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.

Yuyang Nie, Yuanhe Tian, Yan Song, Xiang Ao, and Xiang Wan. 2020. Improving Named Entity Recognition with Attentive Ensemble of Syntactic Information. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4231–4245.

Han Qin, Guimin Chen, Yuanhe Tian, and Yan Song. 2021a. Improving Arabic Diacritization with Regularized Decoding and Adversarial Training. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Han Qin, Guimin Chen, Yuanhe Tian, and Yan Song. 2021b. Improving Federated Leanring for Aspect-based Sentiment Analysis via Topic Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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.

Yan Song, Chunyu Kit, and Xiao Chen. 2009. Transliteration of name entity via improved statistical translation on character sequences. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 57–60, Suntec, Singapore.

Yan Song and Shuming Shi. 2018. Complementary Learning of Word Embeddings. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4368–4374.

Yan Song, Shuming Shi, and Jing Li. 2018a. Joint Learning Embeddings for Chinese Words and Their Components via Ladder Structured Networks. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4375–4381.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018b. Directional Skip-Gram: Explicitly Distinguishing Left and Right Context for Word Embeddings. 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 175–180.

Yan Song and Fei Xia. 2012. Using a Goodness Measurement for Domain Adaptation: A Case Study on Chinese word Segmentation. In *LREC*, pages 3853–3860.

Yan Song, Tong Zhang, Yonggang Wang, and Kai-Fu Lee. 2021. ZEN 2.0: Continue Training and Adaption for N-gram Enhanced Text Encoders. *arXiv preprint arXiv:2105.01279*.

Kai Sun, Richong Zhang, Yongyi Mao, Samuel Mensah, and Xudong Liu. 2020. Relation Extraction with Convolutional Network over Learnable Syntax-Transport Graph. In *AAAI*, pages 8928–8935.

Yuanhe Tian, Guimin Chen, Han Qin, and Yan Song. 2021a. Federated Chinese Word Segmentation with Global Character Associations. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4306–4313, Online.

Yuanhe Tian, Guimin Chen, and Yan Song. 2021b. Aspect-based Sentiment Analysis with Type-aware Graph Convolutional Networks and Layer Ensemble. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2910–2922.

Yuanhe Tian, Guimin Chen, and Yan Song. 2021c. Enhancing Aspect-level Sentiment Analysis with Word Dependencies. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3726–3739.

Yuanhe Tian, Guimin Chen, Yan Song, and Xiang Wan. 2021d. Dependency-driven Relation Extraction with Attentive Graph Convolutional Networks. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Yuanhe Tian, Yan Song, and Fei Xia. 2020a. Joint Chinese Word Segmentation and Part-of-speech Tagging via Multi-channel Attention of Character N-grams. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2073–2084.

Yuanhe Tian, Yan Song, and Fei Xia. 2020b. Supertagging Combinatory Categorial Grammar with Attentive Graph Convolutional Networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6037–6044.

Yuanhe Tian, Yan Song, Fei Xia, and Tong Zhang. 2020c. Improving constituency parsing with span attention. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1691–1703, Online.

Jue Wang and Wei Lu. 2020. Two are better than one: Joint entity and relation extraction with table-sequence encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1706–1721, Online.

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.

Yijun Wang, Changzhi Sun, Yuanbin Wu, Junchi Yan, Peng Gao, and Guotong Xie. 2020. Pre-training entity relation encoder with intra-span and inter-span information. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1692–1705, Online.

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.

Yan Xu, Ran Jia, Lili Mou, Ge Li, Yunchuan Chen, Yangyang Lu, and Zhi Jin. 2016. Improved relation classification by deep recurrent neural networks with data augmentation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1461–1470.

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.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Wei Ye, Bo Li, Rui Xie, Zhonghao Sheng, Long Chen, and Shikun Zhang. 2019. Exploiting Entity BIO Tag Embeddings and Multi-task Learning for Relation Extraction with Imbalanced Data. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1351–1360.

Seunghyun Yoon, Joongbo Shin, and Kyomin Jung. 2018. Learning to rank question-answer pairs using hierarchical recurrent encoder with latent topic clustering. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1575–1584, New Orleans, Louisiana.

Bowen Yu, Mengge Xue, Zhenyu Zhang, Tingwen Liu, Yubin Wang, 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.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation Classification via Convolutional Deep Neural Network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344.

Dongxu Zhang and Dong Wang. 2015. Relation Classification via Recurrent Neural Network. *arXiv preprint arXiv:1508.01006*.

Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. Bidirectional Long Short-Term Memory Networks for Relation Classification. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pages 73–78.

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.

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.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212.

Appendix A. Dataset Statistics

In the experiments, we use two English benchmark datasets for RE, namely, ACE2005EN (ACE05) and SemEval 2010 Task 8 (SemEval) (Hendrickx et al., 2010). For ACE05, we use its English section and follow previous studies (Miwa and Bansal, 2016; Christopoulou et al., 2018; Ye et al., 2019) to pre-process it, where two small subsets cts and un are removed. Then, we split the dataset into training, development, and test sets[10]. For SemEval, we use its official train/test split[11]. The numbers of unique instances are reported in Table 3.

		ACE05	SemEval
# Instances	Train	48,198	8,000
	Dev	11,854	-
	Test	10,097	2,717

Table 3: The number of unique instances (i.e., entity pairs) of ACE05 and SemEval benchmark datasets.

Appendix B. BERT Settings

In our experiments, we use BERT-large with its default settings, i.e., 24 layers with 1024 dimensional hidden-vector and 16 attention heads. For other hyper-parameter settings to train the models, we report them in Table 4. We test all combinations of them for each model and use the one achieving the highest accuracy score in our final experiments.

Hyper-parameters	Values
Learning Rate	$\mathbf{1e-5}, 3e-5, 5e-5$
Warmup Rate	$\mathbf{0.06}, 0.1$
Dropout Rate	$\mathbf{0.1}$
Batch Size	$2, 4, \mathbf{8}$

Table 4: The hyper-parameters tested in tuning our models and the best one used in our final experiments are highlighted in bold.

Appendix C. Model Size and Performance

Table 5 reports the number of trainable parameters and the inference speed (sentences per second) of different models on both ACE05 and SemEval datasets. All models are performed on an NVidia Tesla V100 GPU for training and test.

Models	Para.	ACE05 Speed	SemEval Speed
BERT-large	335,176K	72.18	115.65
+ GCN(Full)	338,333K	55.32	135.85
+ A-GCN(Full)	338,953K	49.23	130.42
+ GCN(N-gram)	344,631K	51.62	155.21
+ A-GCN(N-gram)	345,251K	60.85	147.38

Table 5: The number of trainable parameters (Para.) and inference speed (sentences per second) of the experimented models on the test set of both ACE05 and SemEval datasets. "GCN" is the baseline ; "A-GCN" refers to our approach. "Full" and "N-gram" are the graph construction methods based on all word connections and our approach, respectively.

Appendix D. Experimental Results on the Development Set

Table 6 reports F1 scores of different models on the development set of ACE05.[12]

Appendix E. Mean and Deviation of the Test Results on ACE05 and SemEval

In our experiments, we test models with different configurations. For each model, we train it with the best hyper-parameter setting five times on different random seeds. We report the mean (μ) and standard deviation (σ) of F1 scores on the test sets of both ACE05 and SemEval datasets in Table 7.

[10] We use the train/dev/test splits specified by Miwa and Bansal (2016) at https://github.com/tticoin/LSTM-ER/tree/master/data/ace2005/split

[11] SemEval has only the training and test sets.

[12] SemEval does not have an official dev set.

ID	Models	ACE05
1	BERT-large	77.93
2	+ GCN (Full)	78.01
3	+ A-GCN (Full)	78.22
4	+ GCN (N-gram)	78.15
5	+ A-GCN (N-gram)	**78.42**

Table 6: F1 scores of models with different configurations evaluated on the development set of ACE05, where "Full" and "N-gram" stand for the graph constructed based on all word connections and our approach, respectively. "GCN" is the baseline and "A-GCN" refers to our approach.

	ACE05		SemEval	
	μ	σ	μ	σ
BERT-large	75.96	0.11	88.84	0.13
+ GCN(FULL)	76.73	0.16	88.96	0.12
+ A-GCN(FULL)	77.22	0.14	89.08	0.10
+ GCN(N-gram)	77.13	0.09	89.01	0.08
+ A-GCN (N-gran)	**77.55**	0.16	**89.52**	0.10

Table 7: The mean (μ) and standard deviation (σ) of F1 scores from different models on both ACE05 and SemEval datasets, where "Full" and "N-gram" stand for the graph constructed based on all word connections and our approach, respectively.