Relation Extraction with Self-determined Graph Convolutional Networks

Sunil Kumar Sahu[1], **Derek Thomas**[2], **Billy Chiu**[1], **Neha Sengupta**[1], **Mohammady Mahdy**[1]
[1]Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates
[2]PAX-AI, Abu Dhabi, United Arab Emirates
{sunil.sahu,hon.chiu,neha.sengupta,mohammady.mahdy}@inceptioniai.org
derek.thomas@g42.ai

Abstract

Relation Extraction is a way of obtaining the semantic relationship between entities in text. The state-of-the-art methods use linguistic tools to build a graph for the text in which the entities appear and then a Graph Convolutional Network (GCN) is employed to encode the pre-built graphs. Although their performance is promising, the reliance on linguistic tools results in a non end-to-end process. In this work, we propose a novel model, the Self-determined Graph Convolutional Network (SGCN), which determines a weighted graph using a self-attention mechanism, rather using any linguistic tool. Then, the self-determined graph is encoded using a GCN. We test our model on the TACRED dataset and achieve the state-of-the-art result. Our experiments show that SGCN outperforms the traditional GCN, which uses dependency parsing tools to build the graph.

1 Introduction

Relation extraction (RE) aims at obtaining the semantic relationship between entities using text as a source of knowledge. For instance, from the text snippet, *Steve Jobs and Wozniak co-founded Apple in 1976.*, we can infer that *Steve Jobs* and *Wozniak* have *org:founded_by* relation with *Apple*. RE is an important subtask of information extraction that has significant applications in various higher-order NLP/IR tasks, such as question answering, knowledge graph completion and semantic search (Sarawagi, 2008). Earlier studies on RE were based on feature engineering. Such methods rely on linguistic and lexical tools to obtain the information required for such feature engineering (Zelenko et al., 2003). Additionally, the performance of these methods is hindered by the sparse feature representation used by the models.

With the surge of neural networks, deep learning-based models have become prevalent. In these models, pre-trained word embeddings are employed to solve the feature sparsity problems. Deep learning based RE models can further be categorized along two lines: sequence-based and graph-based models. In sequence-based models, a word sequence is used to embed the text using convolution or recurrent neural networks (Zeng et al., 2014; Zhou et al., 2016). In graph-based models, the text is first converted into a graph using a dependency parser or other linguistic tools and then processed with a graph neural network which encodes neighborhood and feature information. Finally, the encoded graph features are used in RE. Along this line, Liu et al. (2015) and Miwa and Bansal (2016) employed a bidirectional long short-term memory (BiLSTM) network and Zhang et al. (2018) and Guo et al. (2019) employed a graph convolutional network (GCN) (Kipf and Welling, 2017) to encode the textual graph used in their work. Compared to sequence-based models, graph-based models have been shown to be effective in learning long-distance dependencies present in text (Zhang et al., 2018).

Although the state-of-the-art results are obtained using graph-based models, they require external tools to build a graph for the text. Therefore, they are computationally expensive and not fully end-to-end trainable. While sequence-based models do not depend on external linguistic tools, they have been shown less effective for long text, especially when long-distance dependencies are required (Sahu and Anand, 2018). To bridge this gap, we propose a Self-determined GCN (SGCN) which infers (self-determines) a graph for the text using a self-attention mechanism (Vaswani et al., 2017), rather using any external linguistic tool. Then the self-determined graph is encoded using a GCN model. We evaluate the effectiveness of the SGCN on a RE task against several competitive baselines. In summary, our contributions are the following:

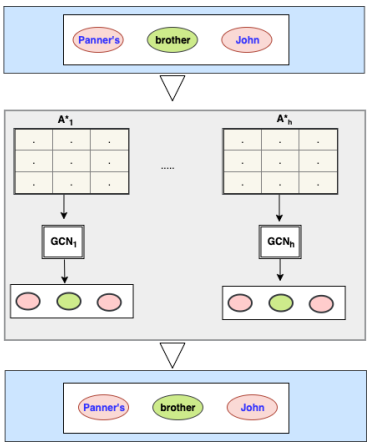

Figure 1: Model architecture of SGCN. First, h adjacency matrices are self-determined from the text using a multi-head self-attention mechanism. Then, a separate GCN is employed for each graph to encode neighborhood information. Finally, the outputs of each GCN are concatenated.

Firstly, we build a novel graph-based model to encode text without the use of any linguistic tools. Secondly, we show the effectiveness of the SGCN model on the RE task and achieve the state-of-the-art performance. Finally, we provide a comprehensive ablation analysis that highlights the importance of SGCN.

2 Graph Convolutional Network (GCN)

The GCN (Kipf and Welling, 2017) is an extension of a convolutional neural network, which encodes neighborhood information in a graph. Let $G = (\mathbf{V}, \mathbf{A}, \mathbf{X})$ be a graph, where $\mathbf{V}$ represents the vertex set and $\mathbf{A} \in \mathbb{R}^{|\mathbf{V}|\times|\mathbf{V}|}$ typically represents a sparse adjacency matrix, where $\mathbf{A}_{(u,v)} = 1$ indicates a connection from node u to node v, else 0, and $\mathbf{X} \in \mathbb{R}^{|\mathbf{V}|\times d}$ represents node embeddings. Each GCN layer takes the node embedding from the previous layer and the adjacency matrix as input and outputs updated node representations. Mathematically, the new node embedding for node $v \in \mathbf{V}$ in the l^{th} layer is:

$$\mathbf{z}_v^{(l+1)} = \sigma\left(\sum_{u=1}^{n} A_{(u,v)}(\mathbf{W}^{(l)}\mathbf{z}_u^{(l)} + \mathbf{b}^{(l)})\right), \quad (1)$$

where $\mathbf{W}^{(l)} \in \mathbb{R}^{d\times o}$ and $\mathbf{b}^{(l)} \in \mathbb{R}^{o}$ are the parameters of the GCN at layer l and σ represents a non-linear activation function.

3 Self-determined Graph Convolution Network (SGCN)

As discussed in Section 1, most current works in NLP use a GCN to encode a pre-built graph, e.g., a dependency parsing graph (Marcheggiani and Titov, 2017) or predicate-argument graph (Marcheggiani et al., 2018). Pre-built graphs require sophisticated tools that have been trained on manual annotations. Although such methods have demonstrated promising results in various NLP tasks, they are computationally expensive, not fully end-to-end trainable and not applicable to low-resource languages. To overcome these issues, our model dynamically self-determines multiple weighted graphs using a multi-head self-attention mechanism (Vaswani et al., 2017) and applies a separate GCN over each one.

Concretely, SGCN represents the words from the text as nodes in a graph and learns multiple adjacency matrices ($\mathbf{A}_1^*, \mathbf{A}_2^* \ldots \mathbf{A}_h^*$), $\mathbf{A}_i^* \in \mathbb{R}^{|\mathbf{V}|\times|\mathbf{V}|}$ in every layer of the GCN (as depicted in Figure 1). Different from the $\mathbf{A}$ used in the traditional GCN, elements in $\mathbf{A}_i^*$ are not binary, but a mean normalized real numbers that represent the strength of the connection in the graph. Mathematically, for the l^{th} layer, we compute the weight of the connection u to v for the i^{th} head, $\mathbf{A}_{i^*_{(u,v)}}$, as:

$$\begin{aligned}\mathbf{M}_{i_{(u,v)}}^{(l+1)} &= \mathbf{ReLU}\left(\frac{\mathbf{K_i}^{(l)}\mathbf{z}_u^{(l)}.(\mathbf{Q_i}^{(l)}\mathbf{z}_v^{(l)})^T}{\sqrt{d}}\right)\\ \mathbf{A}_{i_{(u,v)}}^{*(l+1)} &= \frac{\mathbf{M}_{i_{(u,v)}}^{(l+1)}}{\sum_{u'\in V}\mathbf{M}_{i_{(u',v)}}^{(l+1)}}\end{aligned} \quad (2)$$

where $\mathbf{K_i}^{(l)}, \mathbf{Q_i}^{(l)} \in \mathbb{R}^{d\times d}$ are the trainable parameters. Once all $\mathbf{A_i}^*$s are obtained for the layer, we apply a GCN on each graph to encode the neighborhood information and concatenate the outputs. It is worth mentioning that the attention mechanism used in Eq. 2 differs from the dot-product attention proposed by Vaswani et al. (2017). In this operation, we use the ReLU activation function (Nair and Hinton, 2010) which can mask some of the attention weights by assigning them zero weight. This is more appropriate for the graph, since there are not always mutual connections between every node pair. In contrast to the traditional GCN, which uses the same connections in each layer, the SGCN determines the different connections.

4 RE with SGCNs

For a given text $T = w_1, w_2 \cdots w_n$ and two target entities of interest e_1 and e_2 corresponding to the word (phrase) in T , a RE model takes a triplet (e_1, e_2, T) as input and returns a relation for the pair, (including the no relation category) as output. The set of relations used for inference are predefined. We first transform the text into a sequence of vectors using a pre-trained word embedding. Next, we employ a BiLSTM encoder to capture the context information in the vector sequence, which is then further used to represent the node of the graph.

To further encode the long-distance context, we employ k-layer SGCNs in our model. As explained in Section 3, for each layer, SGCN dynamically determines the weighted connections for the graph using a self-attention mechanism and employs a GCN to propagate neighborhood information into nodes. Next, we employ a layer aggregation, originally proposed by Xu et al. (2018), in which all the SGCN layer outputs, along with a BiLSTM layer output, are concatenated and fed into a feed-forward layer. Finally, for relation classification, we follow Zhang et al. (2018) and employ another feed-forward layer with a softmax operation on the concatenation of the sentence representation and both target entity representations. Sentence and entity representations are obtained by applying max-pooling over the entire sequence and average pooling to the position of entities in the final representation, respectively. Following Zhang et al. (2018) convention, now onward we refer to this model as **C-SGCN**.

5 Experiments and Dataset

We evaluate the performance of the C-SGCN model on the publicly available TACRED RE dataset (Zhang et al., 2017). It is the largest publicly available dataset for sentence-level RE. TACRED is manually annotated with 41 categories of relations between subjects and objects. While the subject of these relations is PERSON and ORGANISATION, object consist of 16 fine-grained entity types that include: DATE, LOCATION, TIME, etc. The dataset has 68124, 22631 and 15509 instances for training, development and test out of which 79.5% of the instances are labeled as no_relation.

We employ the entity masking strategy to preprocess the dataset (Zhang et al., 2017, 2018), where each subject (and object similarly) will be replaced with a special $Subj- < NER >$ token.

Model	P	R	F1 (%)
Patterns∗ (Angeli et al., 2015)	86.9	23.2	36.6
LR* (Surdeanu et al., 2012)	73.5	49.9	59.4
LR + Patterns∗ (Angeli et al., 2015)	72.9	51.8	**60.5**
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 (Zhang et al., 2018)	69.9	63.3	66.4
S-GCN (Wu et al., 2019)	-	-	67.0
C-AGGCN (Guo et al., 2019)	69.6	66.0	**67.8**
CNN∗ (Kim, 2014)	75.6	47.5	58.3
CNN-PE∗ (Zeng et al., 2014)	70.3	54.2	61.2
LSTM∗ (Zhang and Wang, 2015)	65.7	59.9	62.7
PA-LSTM (Zhang et al., 2017)	65.7	64.5	65.1
C-SGCN-Softmax	69.3	65.4	67.3
C-SGCN	69.8	65.9	**67.8**

Table 1: Performance comparison of SGCN models against baselines. ∗ refers the performance was reported by Zhang et al. (2017). The performances of the feature-based, sequence-based and graph-based model are separated in the first, second and third part of the table. The best F1 score in each section is highlighted.

For instance, *"$MetLife_{Obj}$ says it acquires AIG unit $ALICO_{Subj}$ for 15.5 billion dollars"* will become *"Obj-Org says it acquires AIG unit Subj-Org for 15.5 billion dollars"*. Similar to other works (Zhang et al., 2017, 2018; Guo et al., 2019), we employed PoS tag embedding, and entity tag embedding[1] along with word embedding to represent a word in the input of C-SGCN.

5.1 Training and Hyper-parameter Settings

In our model, ReLU activation function is employed in all GCN operation. We used 300 dimension GloVe vector to initialize word embeddings and 30 dimension random vectors to initialize PoS and entity tag embeddings. Parameters of the models are optimized using the stochastic gradient decent with batch size 50 and initial learning rate of 0.3. We used early stopping with patience equal to 5 epochs in order to determine the best training epoch. For other hyper-parameters, we perform a non-exhaustive hyper-parameter search based on the development set of the dataset. The dimension of SGCN and LSTM layer is set to 300. We used 2 layer SGCNs with 3 heads in each for our experiments. To prevent overfitting, we used dropout (Srivastava et al., 2014) in SGCN and LSTM layers with dropout rate equals to 0.5. The remaining hyperparameter values are adopted from Zhang et al. (2018).

[1]PoS and entity tags are provided with the dataset

5.2 Baseline Models

We compare our C-SGCN model against several competitive baselines, which include feature engineering-based methods (Surdeanu et al., 2012; Angeli et al., 2015), sequence-based methods (Zeng et al., 2014; Zhang and Wang, 2015; Zhang et al., 2017) and graph-based methods (Xu et al., 2015; Tai et al., 2015; Zhang et al., 2018; Wu et al., 2019; Guo et al., 2019). Apart from these, we additionally prepare *C-SGCN-Softmax*, uses C-SGCN model with softmax to compute weighted graph in SGCN. To avoid any effects from the external enhancements, we don't consider methods that use BERT (Devlin et al., 2019) or any other language model as pre-training in their models. We leave these experiments for future work.

5.3 Performance Comparison

Table 1 shows the performance comparison of SGCN models against all baselines. From the table, we can observe that C-SGCN outperforms all the feature-based and sequence-based models by a noticeable margin. Furthermore, compared to graph-based models, C-SGCN outperforms SDP-LSTM (Xu et al., 2015), Tree-LSTM (Tai et al., 2015), C-GCN (Zhang et al., 2018) and S-GCN (Wu et al., 2019). However, C-SGCN's performance is same as C-AGGCN (Guo et al., 2019) in terms of F1 score. It is worth mentioning that all of these works (except C-SGCN) employ a dependency parser to build the graph for the text, which is further encoded using a graph neural network. A dependency parsing requires external tool which is computationally expensive and time consuming. In addition to this, our model C-SGCN performance is 0.5 points higher than *C-SGCN-Softmax* in terms of F1 score, verified the claim that use of ReLU activation function for computing edge weight is more appropriate for GCN framework.

5.4 Ablation Study

We have demonstrated the strong empirical results obtained by the C-SGCN model. Next, we want to understand the contribution of each component employed in the model. We conduct an ablation test by removing some of these components. The ablated models are: (a) **No_SGCN** is the C-SGCN model without the SGCN component, i.e., the sentence representation and entity representations are obtained from the output of the BiLSTM layer. (b) **No_LSTM**, represents the C-SGCN model without BiLSTM. Finally, (c) **No_LA**, represents the C-SGCN model without the layer aggregation, i.e., the last layer output of SGCN is used to obtain the sentence and entity representations.

Model	P	R	F1 (%)
C-SGCN	69.8	65.9	**67.8**
No_SGCN	71.7	61.6	66.3
No_LSTM	68.5	43.9	53.5
No_LA	75.2	55.9	64.1

Table 2: Ablation analysis on the test set of TACRED dataset.

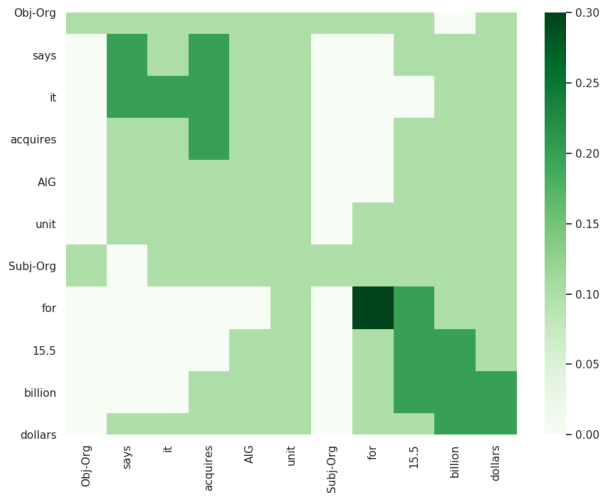

Figure 2: Heat map of the adjacency matrix determined by SGCN in first layer. The sentence is 'Obj-Org says it acquires AIG unit Subj-Org for 15.5 billion dollars'. Heat map values are approximated to one decimal point.

Table 2 shows the results of the ablation study performance. From the table, we can observe that all the components employed in C-SGCN models have a noticeable contribution to the overall all performance. In particular, the performance of No_SGCN is 1.5 points lower than the C-SGCN in terms of F1 score, demonstrating the strong contribution of SGCN.

5.5 Interpretation of the Self-determined Graph

The proposed approach dynamically determines the weighted graph for the text using a self-attention mechanism. In this section, we try to visualise the graph by plotting a heat map of the adjacency matrix obtained by the model. We wish to examine whether the SGCN indeed learns the connections that are important for the relation extraction task. Figure 2 depicts the heat map figure of sentence

"Obj-Org says it acquires AIG unit Subj-Org for 15.5 billion dollars". The sentence expresses the *org:parents* relation between subject and object. From the figure, one can observe that the SGCN can infer a connection from the target entities to most of the other words in the text. In addition to this, it also inferring a strong self-connection weight to the words that are important for the prediction of the relation. Finally, the connections in the inferred graph are not symmetric.

6 Related Work

RE is a well-studied field of knowledge extraction. Traditional feature-based methods rely on manual features obtained from various tools and lexical resources (Culotta and Sorensen, 2004). Recently, various neural network based methods have also been applied for RE tasks. These include convolutional neural networks (Zeng et al., 2014), recurrent neural networks (Zhou et al., 2016), transformer networks (Verga et al., 2018) and graph-based networks, e.g., Graph LSTMs (Peng et al., 2017) and GCNs (Zhang et al., 2018; Wu et al., 2019; Guo et al., 2019). Guo et al. (2019), Wu et al. (2019) and Zhang et al. (2018) employed a GCN to encode the dependency graph of the text. In their works, dependency graph for the text is obtained using linguistic tools. However, we employ a GCN to encode a self-determined graph, obtained automatically using a self-attention mechanism.

GCNs have been studied in various domains, using a variety of graphs, e.g., social network graphs (Kipf and Welling, 2017), chemical reaction network graphs (Coley et al., 2019) etc. In text, GCNs are employed to encode non-local dependencies present between the words in a text. They have been successfully used in co-occurrence graphs (Yao et al., 2019), predicate-argument graphs (Marcheggiani et al., 2018), dependency parsing graphs (Zhang et al., 2018) and heterogeneous graph (Sahu et al., 2019). To the best of our knowledge, this is the first work that employs a GCN on a fully self-determined graph.

7 Conclusion and Future Works

In this work, we proposed a novel model, C-SGCN, for the RE task. Our model dynamically determines the graph for the text using a self-attention mechanism. Although the proposed model is evaluated on the RE task, it is generic and can be applied for other tasks. Experimental results show that our model achieves comparable performance to the state-of-the-art neural models that uses dependency parsing tool to obtain a graph for the text.

Recently, several studies have demonstrated that employment of a pre-trained language model in end-to-end neural models further improves the performance of the downstream task, in future, we will try to incorporate a pre-trained language model with proposed C-SGCN model and improve the performance of RE. Besides, the applicability of SGCN in other text mining tasks are yet to be validated.

References

Gabor Angeli, Victor Zhong, Danqi Chen, Arun Tejasvi Chaganty, Jason Bolton, Melvin Jose Johnson Premkumar, Panupong Pasupat, Sonal Gupta, and Christopher D. Manning. 2015. Bootstrapped self training for knowledge base population. In *Proceedings of the Text Analysis Conference*.

Connor W Coley, Wengong Jin, Luke Rogers, Timothy F Jamison, Tommi S Jaakkola, William H Green, Regina Barzilay, and Klavs F Jensen. 2019. A graph-convolutional neural network model for the prediction of chemical reactivity. *Chemical science*, 10(2):370–377.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the Annual Meeting on Association for Computational Linguistics*, pages 423–430.

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 North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

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

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations*.

Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and Houfeng Wang. 2015. A dependency-based neural network for relation classification. In *Proceedings of the Annual Meeting of the Association*

for Computational Linguistics and the International Joint Conference on Natural Language Processing, pages 285–290.

Diego Marcheggiani, Joost Bastings, and Ivan Titov. 2018. Exploiting semantics in neural machine translation with graph convolutional networks. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 486–492.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1105–1116.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the International Conference on Machine Learning*, pages 807–814.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph lstms. *Transactions of the Association for Computational Linguistics*, 5:101–115.

Sunil Kumar Sahu and Ashish Anand. 2018. Drug-drug interaction extraction from biomedical texts using long short-term memory network. *Journal of Biomedical Informatics*, 86:15 – 24.

Sunil Kumar Sahu, Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2019. Inter-sentence relation extraction with document-level graph convolutional neural network. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 4309–4316.

Sunita Sarawagi. 2008. Information extraction. *Foundations and Trends in Databases*, 1(3):261–377.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465.

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 Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, pages 1556–1566.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 872–884.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *Proceedings of the International Conference on Machine Learning*, pages 6861–6871.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*, pages 5449–5458.

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 Conference on Empirical Methods in Natural Language Processing*, pages 1785–1794.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7370–7377.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *J. Mach. Learn. Res.*, 3:1083–1106.

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

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 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 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 Annual Meeting of the Association for Computational Linguistics*, pages 207–212.