Dependency-driven Relation Extraction with Attentive Graph Convolutional Networks

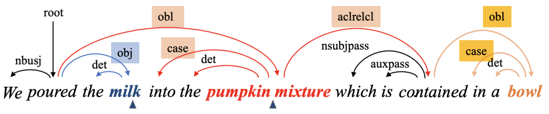
Yuanhe Tian[♥*], **Guimin Chen**[♢*], **Yan Song**[♠♡†], **Xiang Wan**[♡]
[♥]University of Washington [♢]QTrade
[♠]The Chinese University of Hong Kong (Shenzhen)
[♡]Shenzhen Research Institute of Big Data
[♥]`yhtian@uw.edu` [♢]`chenguimin@foxmail.com`
[♠]`songyan@cuhk.edu.cn` [♡]`wanxiang@sribd.cn`

Abstract

Syntactic information, especially dependency trees, has been widely used by existing studies to improve relation extraction with better semantic guidance for analyzing the context information associated with the given entities. However, most existing studies suffer from the noise in the dependency trees, especially when they are automatically generated, so that intensively leveraging dependency information may introduce confusions to relation classification and necessary pruning is of great importance in this task. In this paper, we propose a dependency-driven approach for relation extraction with attentive graph convolutional networks (A-GCN). In this approach, an attention mechanism upon graph convolutional networks is applied to different contextual words in the dependency tree obtained from an off-the-shelf dependency parser, to distinguish the importance of different word dependencies. Consider that dependency types among words also contain important contextual guidance, which is potentially helpful for relation extraction, we also include the type information in A-GCN modeling. Experimental results on two English benchmark datasets demonstrate the effectiveness of our A-GCN, which outperforms previous studies and achieves state-of-the-art performance on both datasets.[1]

1 Introduction

Relation extraction (RE), which aims to detect the relationship between entity mentions from raw text, is one of the most important tasks in information extraction and retrieval, and plays a crucial role in supporting many downstream natural language processing (NLP) applications such as text mining (Distiawan et al., 2019), sentiment analysis (Sun et al., 2019), question answering (Xu et al., 2016a), and summarization (Wang and Cardie, 2012).


root
obl
aclrelcl
obl
nbusj
obj
det
case
det
nsubjpass
auxpass
case
det
We poured the milk into the pumpkin mixture which is contained in a bowl
Figure 1: An illustration of noises in the dependency tree that can hurt relation extraction, where the word dependency connected in between "*pumpkin mixture*" and "*bowl*" (whose relation is *content-container*) may introduce confusion to the system when the object is to predict the relation between "*milk*" and "*pumpkin mixture*" (whose relation is *entity-destination*).

Recently, neural RE methods (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) with powerful encoders (such as CNN, RNN, and Transformers) have significantly improved model performance for RE without requiring any elaborately designed systems or manually constructed features. These methods are superior in capturing contextual information and thus enable RE systems to better understand the text and identify relations between entities in the given text. Adopting neural models to help RE is not only straightforward and effective, but is also expected to incorporate more diverse and informative knowledge into RE systems. Among all different knowledge sources, syntactic information, especially the dependency trees, have been demonstrated to be beneficial in many studies (Miwa and Bansal, 2016; Zhang et al., 2018; Sun et al., 2020; Chen et al., 2021) because they provide long-distance word connections between useful words and thus accordingly guide the system to better extract relations between entity pairs.

However, intensively leveraging dependency information may not always lead to good RE performance, because the noise in the dependency tree can potentially introduce confusions to relation classification (Xu et al., 2015; Yu et al., 2020),

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

especially when those trees are automatically generated. For example, Figure 1 shows an example sentence with its dependency tree, where the dependency connection between "*pumpkin mixture*" and "*bowl*" may introduce noise when the object is to predict the relation between "*milk*" and "*pumpkin mixture*". Therefore, previous studies have always required necessary pruning strategies before encoding the dependency information through a particular model such as LSTM (Xu et al., 2015) or graph convolutional networks (GCN) (Zhang et al., 2018). Because fixed pruning strategies are not guaranteed to result in a sub-tree with all important contextual information included and with all noise filtered out, it is necessary to design an appropriate way for distinguishing the noise in the dependency tree and modelling them accordingly.

In this paper, we propose a dependency-driven neural approach for RE, where attentive graph neural network (A-GCN) is proposed to distinguish the important contextual information for this task. Furthermore, given that the dependency types (e.g., nominal subject) that associate with dependency connections are also potentially useful for RE since they contain the syntactic instruction among connected words, we further improve A-GCN by introducing type information into it. Specifically, we first obtain the dependency tree of an input sentence from an off-the-shelf toolkit, then build the graph over the dependency tree, and assign different weights to different labeled dependency connections between any two words, with the weights computed based on the connections and their dependency types, lastly predict relations by the A-GCN according to the learned weights. In doing so, not only is A-GCN able to distinguish important contextual information from dependency trees and leverage them accordingly, such that reliance on pruning strategies is unnecessary, but A-GCN can also leverage the dependency type information that is omitted by most previous studies (in particular, the studies that also use attention mechanism (Guo et al., 2019)). Experimental results on two English benchmark datasets, i.e., ACE2005EN and SemEval 2010 Task 8, demonstrate the effectiveness of our approach to RE through A-GCN equipped with dependency type information. State-of-the-art performance is observed on both datasets.

2 The Proposed Approach

RE is conventionally performed as a typical classification task. Our approach follows this paradigm by using A-GCN and incorporates dependency information to improve model performance, where the overall architecture of our model is illustrated in Figure 2. Specifically, given an unstructured input sentence $\mathcal{X} = x_1, \cdots, x_n$ with n words and let E_1 and E_2 denote two entities in $\mathcal{X}$, our approach predicts the relation $\widehat{r}$ between E_1 and E_2 by

$$\widehat{r} = \underset{r \in \mathcal{R}}{\arg\max}\ p\left(r | \text{A-GCN}\left(\mathcal{X}, \mathcal{T}_{\mathcal{X}}\right)\right) \qquad (1)$$

where $\mathcal{T}_{\mathcal{X}}$ is the dependency tree of $\mathcal{X}$ obtained from an off-the-shelf toolkit, $\mathcal{R}$ is the relation type set; p computes the probability of a particular relation $r \in \mathcal{R}$ given the two entities and $\widehat{r}$ the output of A-GCN, which takes $\mathcal{X}$ and $\mathcal{T}_{\mathcal{X}}$ as the input. Following texts start with a brief introduction of the standard GCN model, then elaborate our proposed A-GCN equipped with dependency type information, and lastly illustrate the process of applying A-GCN to the classification paradigm for RE.

2.1 Standard Graph Convolutional Networks

Generally, a good text representation is a prerequisite to achieve outstanding model performance (Song et al., 2017; Bojanowski et al., 2017; Song et al., 2018; Song and Shi, 2018; Hajdik et al., 2019). To enhance the text representation and thus obtain a good understanding of the running text, many studies (Song et al., 2009, 2012; Song and Xia, 2013; Xu et al., 2015; Miwa and Bansal, 2016; Zhang et al., 2019; Mandya et al., 2020; Nie et al., 2020) tried to leverage contextual features, such as n-grams and syntactic information, through different model architectures. Among all these architecture choices, graph convolutional networks (GCN) is a widely used architecture to encode the information in a graph, where in each GCN layer, information in each node communicates to its neighbors through the connections between them. The effectiveness of GCN models to encode the contextual information over a graph of an input sentence has been demonstrated by many previous studies (Zhang et al., 2018; Guo et al., 2019; Sun et al., 2020; Chen et al., 2020; Yu et al., 2020; Mandya et al., 2020; Tian et al., 2020c, 2021a). Normally, the graph in the standard GCN model is built from word dependencies and is represented by an adjacency matrix $\mathbf{A} = (a_{i,j})_{n \times n}$ where $a_{i,j} = 1$ if $i = j$ or there is a dependency connection[2] (arc) between two words x_i and x_j in the dependency tree $\mathcal{T}_{\mathcal{X}}$ and $a_{i,j} = 0$ otherwise. Based on $\mathbf{A}$, for

[2]Normally the direction of the connection is ignored.

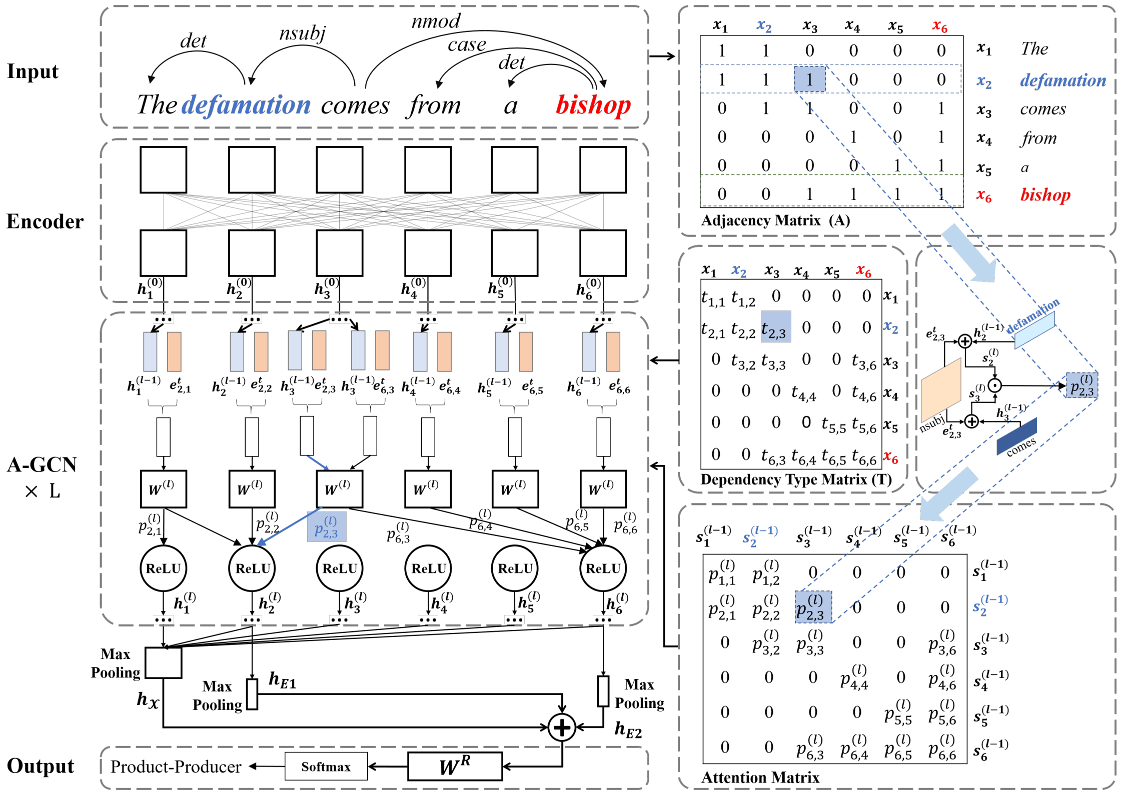


Figure 2: The overall architecture of the proposed A-GCN for RE illustrated with an example input sentence (the two entities "*defamation*" and "*bishop*" are highlighted in blue and red colors, respectively) and its dependency tree. The left part shows our A-GCN model where the attention weights are applied to different connections to model the dependency type-aware contextual information. The right part illustrates the adjacency matrix $\mathbf{A}$ for the dependency graph and the process to compute the attention weights (i.e., $p_{i,j}^{(l)}$) for different connections.

each word $x_i \in \mathcal{X}$, the l-th GCN layer gathers the information carried by its context words in $\mathcal{T_X}$ and computes the output representation $\mathbf{h}^{(l)_i}$ for x_i by:

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

where $\mathbf{h}_j^{(l-1)}$ denotes the output representation of x_j from the (l-1)-th GCN layer[3], $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are trainable matrices and the bias for the l-th GCN layer, respectively, and σ is the $ReLU$ activation.

2.2 A-GCN with Dependency Type

It is noted that in standard GCN (e.g., Eq. (2)), the connections among words are treated equally (i.e., $a_{i,j}$ is either 0 or 1). Therefore, GCN-based models for RE are not able to distinguish the importance of different connections and thus pruning on them is of great importance for RE. Therefore, we propose A-GCN for this task, which uses an attention mechanism to compute the weights for different connections so that the model is able to leverage different dependency connections accordingly. In addition, the standard GCN and most previous studies omit the dependency types associated with the dependency connections, where those types contain highly useful information for RE and are introduced into A-GCN in this work. Specifically, we firstly represent dependency types in $\mathcal{T_X}$ by a type matrix $\mathbf{T} = (t_{i,j})_{n \times n}$, where $t_{i,j}$ is the dependency type (e.g., *nsubj*) associated with the directed dependency connection[4] between x_i and x_j. Next, we map each type $t_{i,j}$ to its embedding $\mathbf{e}_{i,j}^{t}$. Then, at the l-th GCN layer, the weight for the connection between x_i and x_j is computed by

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

where $a_{i,j} \in \mathbf{A}$, "$\cdot$" denotes inner production, and $\mathbf{s}_i^{(l)}$ and $\mathbf{s}_i^{(l)}$ are two intermediate vectors for x_i and

[3]$\mathbf{h}_j^{(0)}$ is the output of the encoder for x_j.

[4]It means $t_{i,j}$ and $t_{j,i}$ are represented in different dependency types to model directions of connections between x_i and x_j. For example, if $t_{i,j}$ is *nsubj*, then $t_{j,i}$ is *#nsubj*.

x_j, respectively, which are computed by

$$\mathbf{s}_i^{(l)} = \mathbf{h}_i^{(l-1)} \oplus \mathbf{e}_{i,j}^t \tag{4}$$

and

$$\mathbf{s}_j^{(l)} = \mathbf{h}_j^{(l-1)} \oplus \mathbf{e}_{i,j}^t \tag{5}$$

with "$\oplus$" denoting the vector concatenation operation. Afterwards, we apply the weight $p_{i,j}^{(l)}$ to the associated dependency connection between x_i and x_j and obtain the output representation of x_i by

$$\mathbf{h}_i^{(l)} = \sigma \left(\sum_{j=1}^{n} p_{i,j}^{(l)} \left(\mathbf{W}^{(l)} \cdot \widetilde{\mathbf{h}}_j^{(l-1)} + \mathbf{b}^{(l)} \right) \right) \tag{6}$$

with σ, $\mathbf{W}^{(l)}$, and $\mathbf{b}^{(l)}$ following the same notations in Eq. (2) for standard GCN, and $\widetilde{\mathbf{h}}_j^{(l-1)}$ (a type-enhanced representation for x_j) computed by

$$\widetilde{\mathbf{h}}_j^{(l-1)} = \mathbf{h}_j^{(l-1)} + \mathbf{W}_T^{(l)} \cdot \mathbf{e}_{i,j}^t \tag{7}$$

where $\mathbf{W}_T^{(l)}$ maps the dependency type embedding $\mathbf{e}_{i,j}^t$ to the same dimension as $\mathbf{h}_j^{(l-1)}$.

Compared with standard GCN (i.e., Eq. (2)), our approach uses numerical weighting (i.e., $p_{i,j}^{(l)} \in [0, 1]$) rather than a binary choice for $a_{i,j}$, to distinguish the importance of different connections so as to leverage them accordingly. In addition, we integrate the dependency type information into both the computed weight (i.e., $p_{i,j}^{(l)}$) and the output representation of x_i (i.e., $\mathbf{h}_i^{(l)}$), which is not considered in most previous studies.

2.3 Relation Extraction with A-GCN

Before applying A-GCN for RE, we firstly encode the input $\mathcal{X}$ into hidden vectors by BERT (Devlin et al., 2019) with $\mathbf{h}_i^{(0)}$ denoting the hidden vector for x_i. Next, we feed $\mathbf{h}_i^{(0)}$ to our proposed A-GCN model with L layers and obtain the corresponding output $\mathbf{h}_i^{(L)}$. Then, we apply the max pooling mechanism to two text spans: the first is on all $\mathbf{h}_i^{(L)}$ to obtain the global sentence representation $\mathbf{h}_{\mathcal{X}}$ by

$$\mathbf{h}_{\mathcal{X}} = \text{MaxPooling}(\{\mathbf{h}_1^{(L)}, \cdots, \mathbf{h}_n^{(L)}\}) \tag{8}$$

and the second is on $\mathbf{h}_i^{(L)}$ of those words that belongs to an entity mention (i.e., $E_k, k = 1, 2$) to compute the representation for entity $\mathbf{h}_{E_k}$ by

$$\mathbf{h}_{E_k} = \text{MaxPooling}(\{\mathbf{h}_i^{(L)} | x_i \in E_k\}) \tag{9}$$

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

$$\mathbf{o} = \mathbf{W}_R \cdot (\mathbf{h}_{\mathcal{X}} \oplus \mathbf{h}_{E_1} \oplus \mathbf{h}_{E_2}) \tag{10}$$

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

$$\widehat{r} = \arg\max \frac{\exp(o^u)}{\sum_{u=1}^{|\mathcal{R}|} \exp(o^u)} \tag{11}$$

with $\mathbf{o}^u$ representing the value at dimension u in $\mathbf{o}$.

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

Table 1: The number of unique instances (i.e., entity pairs) of ACE05 and SemEval benchmark datasets.

3 Experimental Settings

3.1 Datasets

In the experiments, we use two English benchmark datasets for RE, namely, ACE2005EN (ACE05)[5] and SemEval 2010 Task 8 (SemEval)[6] (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 (two small subsets cts and un are removed) and split the documents into training, development, and test sets[7]. For SemEval, we use its official train/test split[8]. The numbers of unique relation types in ACE05 and SemEval are 7 and 19, respectively. We report the number of instances (i.e., entity pairs), for train/dev/test sets of ACE05 and SemEval benchmark datasets in Table 1.

3.2 Dependency Graph Construction

To construct graphs for A-GCN, we use Standard CoreNLP Toolkits (SCT)[9] to obtain the dependency tree $\mathcal{T}_{\mathcal{X}}$ for each input sentence $\mathcal{X}$. Although our approach is able to distinguish the importance of different dependency connections through the attention mechanism, it is still beneficial if we can filter

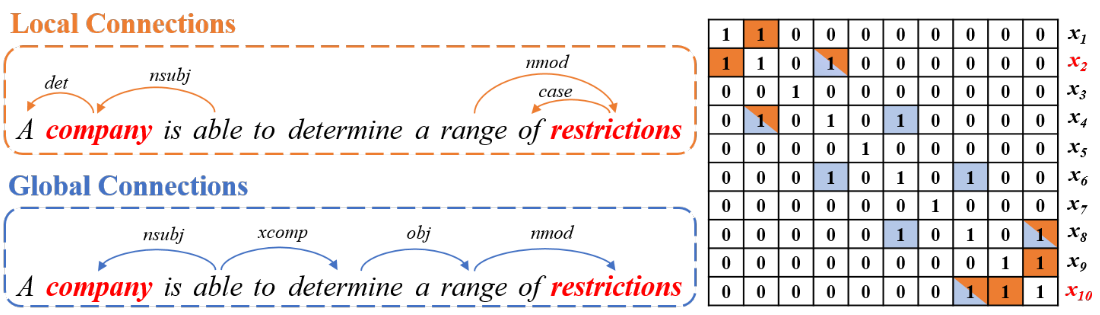
[5]We obtain the official data (LDC2006T06) from `https://catalog.ldc.upenn.edu/LDC2006T06`.

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

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

[8]SemEval only includes the training and test sets.

[9]We download the version 3.9.2 from `https://stanfordnlp.github.io/CoreNLP/`.


Local Connections
det
nsubj
nmod
case
A company is able to determine a range of restrictions
Global Connections
nsubj
xcomp
obj
nmod
A company is able to determine a range of restrictions

$$\begin{array}{cccccccccc|c}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_1 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & x_2 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_3 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & x_4 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & x_5 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & x_6 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & x_7 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & x_8 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & x_9 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & x_{10}
\end{array}$$

Figure 3: An illustration on the two (i.e., local and global) groups of dependency connections for an example sentence (entities are highlighted in red color) with an adjacency matrix (on the right) built upon all connections from the two groups. Local and global connections are represented in orange and blue colors, respectively,

out those dependency connections that bring confusions to RE through particular pruning strategies. Motivated by previous studies (Xu et al., 2015; Zhang et al., 2018; Yu et al., 2020), in this paper, we construct the graph for A-GCN by including two groups of dependency connections, namely, the *local connections* and the *global connections*. In detail, local connections include all dependencies that directly connect to the heads of two entities and global connections include all dependencies along the shortest dependency path (SDP) between the head of two entities, where in many cases words that do not directly connected to the two entities are also involved. With an example sentence including two entities (i.e., "*company*" and *benchmarking*), Figure 3 illustrates the two groups of dependency connections and the resulted adjacency matrix, which is built with the connections from the two groups[10]. It is worth noting that, when the SDP is short, there might be more connections in the local group than that in the global one.

3.3 Implementation

Following Soares et al. (2019), we insert four special tokens (i.e., "*<e1>*", "*</e1>*", "*<e2>*", and "*</e2>*") into the input sentence to mark the boundary[11] of the two entities to be investigated, which allows the encoder to distinguish the position of entities during encoding and thus improves model performance. For the encoder, we try BERT (Devlin et al., 2019), because it is a powerful pre-trained language model which and whose variants have achieved state-of-the-art performance in many NLP tasks (Wu and He, 2019; Soares et al., 2019; Wu et al., 2019; Diao et al., 2020; Song et al., 2020; Antoun et al., 2020; Tian et al., 2020a,b,d, 2021b; Qin et al., 2021; Song et al., 2021). Specifically, we use the uncased version of BERT-base and BERT-large[12] following the default settings (e.g., for BERT-base, we use 12 layers of multi-head attentions with 768-dimensional hidden vectors; for BERT-large, we use 24 layers of multi-head attentions with 1024-dimensional hidden vectors). For A-GCN, we randomly initialize all trainable parameters and the dependency type embeddings. For evaluation, we follow previous studies to use the standard micro-F1 scores[13] for ACE05 and use the macro-averaged F1 scores[14] for SemEval. In our experiments, we try different combinations of hyper-parameters, and tune them on the dev set, then evaluate on the test set by the model that achieves the highest F1 score on the dev set.[15]

4 Results

4.1 Overall Results

In the experiments, we run our A-GCN models using BERT-base and BERT-large encoder on graphs with and without applying dependency pruning strategies, which correspond to the graph built upon the combined local and global connections ("L + G"), as well as the one constructed by the full dependency graph ("Full"), respectively. We also run baselines with standard GCN and standard graph attentive networks (GAT) (Veličković et al., 2017) with the same graph. For both standard GCN and A-GCN, we try different numbers of layers (i.e. 1 to 3

[10]We do not distinguish the two groups of connections in A-GCN once they are represented by the adjacency matrix.

[11]For example, "*<e1>*" and "*</e1>*" are respectively inserted right before and after the entity E_1 in the input $\mathcal{X}$.

[12]We download different BERT models from `https://github.com/huggingface/transformers`.

[13]We use the evaluation script from *sklearn* framework.

[14]We use the official evaluation script downloaded from `http://semeval2.fbk.eu/scorers/task08/SemEval2010_task8_scorer-v1.2.zip`.

[15]We report the hyper-parameter settings of different models with their size and running speed in Appendix A and B.

ID	Models	ACE05	SemEval
1	BERT-base	75.31	87.87
2	+ GAT (Full)	76.16	88.39
3	+ GAT (L + G)	75.79	88.53
4	+ 1 GCN layer (Full)	74.91	87.58
5	+ 1 A-GCN layer (Full)	76.63	88.34
6	+ 1 GCN layer (L + G)	75.51	88.64
7	+ 1 A-GCN layer (L + G)	77.10	89.03
8	+ 2 GCN layers (Full)	75.09	88.66
9	+ 2 A-GCN layers (Full)	77.25	88.70
10	+ 2 GCN layers (L + G)	76.11	88.62
11	+ 2 A-GCN layers (L + G)	**77.30**	**89.16**
12	+ 3 GCN layers (Full)	75.69	88.54
13	+ 3 A-GCN layers (Full)	76.26	88.63
14	+ 3 GCN layers (L + G)	76.85	88.33
15	+ 3 A-GCN layers (L + G)	76.36	88.70

(a) BERT-base

ID	Models	ACE05	SemEval
1	BERT-large	76.79	89.02
2	+ GAT (Full)	78.25	89.39
3	+ GAT (L + G)	78.71	89.44
4	+ 1 GCN layer (Full)	77.63	88.98
5	+ 1 A-GCN layer (Full)	78.53	89.54
6	+ 1 GCN layer (L + G)	77.49	89.11
7	+ 1 A-GCN layer (L + G)	78.48	89.69
8	+ 2 GCN layers (Full)	78.67	89.43
9	+ 2 A-GCN layers (Full)	78.91	89.70
10	+ 2 GCN layers (L + G)	78.82	89.42
11	+ 2 A-GCN layers (L + G)	**79.05**	**89.85**
12	+ 3 GCN layers (Full)	78.08	89.62
13	+ 3 A-GCN layers (Full)	78.45	89.46
14	+ 3 GCN layers (L + G)	78.64	89.19
15	+ 3 A-GCN layers (L + G)	78.83	89.56

(b) BERT-large

Table 2: F1 scores of our A-GCN models and the baselines (i.e., BERT-only, standard GAT, and standard GCN) under different settings with BERT-base (a) and BERT-large (b) used. All graph-based models (i.e., GAT, GCN, and A-GCN) are tested with two settings: the first is using the full graph (Full) with all dependency connections involved and the second is using the combination of local and global connections (L + G). We also run GCN and A-GCN with different numbers of layers (i.e., 1 to 3 layers) for fair comparisons.

layers). In addition, we try BERT-base and BERT-large baselines without using any dependency information. Table 2 shows the F1 scores of our A-GCN models and all the aforementioned baselines on the test set of ACE05 and SemEval.[16]

There are several observations. First, A-GCN functions well when using BERT-base or BERT-large as encoder, where the consistent improvement is observed over the BERT-only baselines (ID: 1) across two benchmark datasets, even though the BERT baselines have already achieve good performance. Second, for both datasets, A-GCN outperforms GAT (ID: 2, 3) and standard GCN baselines (ID: 4, 6, 8, 10, 12, 14) with the same graph (i.e., either "L + G" or "Full") and equal number of layers. Particularly, when full dependency graph is used, it is noted that, in some cases (e.g., ID: 8 for BERT-base on ACE05), standard GCN obtains very limited improvements (or even worse results) over the BERT-only baseline (ID: 1), whereas our A-GCN models (e.g., ID: 9 for BERT-base) is able to consistently outperform the BERT-only baseline and achieve higher performance. We attribute this observation to the attention mechanism used to weigh different dependency connections, which allows A-GCN to distinguish the noise in the graph and thus leverage useful dependency information accordingly. Third, among the models with different numbers of A-GCN layers, the ones (e.g., ID: 11 for BERT-base and ID: 11 for BERT-large) with two A-GCN layers achieves the highest scores, where similar tread is observed from the standard GCN baselines. Besides, we find that our A-GCN models (as well as the standard GCN baselines) with the local and global connections (i.e., "L + G") consistently outperform the ones with full dependency graph (i.e., "Full"). These observations are relatively intuitive since the dependency information may introduce more noise to RE when it is leveraged in an intensive way (e.g., by using more layers or the full dependency tree without pruning).

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

4.2 Comparison with Previous Studies

In addition, we compare our best models (with "L + G" or "Full" graphs) using BERT-large encoder and two A-GCN layers (ID: 9 and 11) with previous studies. The test results (F1 scores) are reported in Table 3, where our model with both local and global connections (i.e., "L + G") outperforms all previous studies and achieves state-of-the-art performance on the two benchmark datasets. Specifically, compared with Guo et al. (2019) who proposed an graph-based approach with attentions to leverage dependency connections, our approach leverages both dependency connections and dependency types among all input words and thus provides a better way to comprehensively leverage the dependency information. In addition, although Mandya et al. (2020) proposed an approach to leverage both dependency connections and dependency types through attentions, they added the dependency type directly to the input word embeddings along with POS embeddings, and the attention in

MODELS	ACE05	SEMEVAL
XU ET AL. (2015)	-	83.7
WANG ET AL. (2016)	-	88.0
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
GUO ET AL. (2019)	-	85.4
SUN ET AL. (2020)	-	86.0
MANDYA ET AL. (2020)	-	85.9
YU ET AL. (2020)	-	86.4
A-GCN (BERT) (FULL)	78.91	89.70
A-GCN (BERT) (L + G)	**79.05**	**89.85**

Table 3: The comparison (F1 scores) between previous studies and our best models using two A-GCN layers and BERT-large encoder on ACE05 and SemEval.

their approach is a separate stand-alone module which is added on the top of the GCN layer. On the contrary, in our approach, the dependency type is added to each A-GCN layer and the attention mechanism is directly applied to each dependency connection in the A-GCN layer. Therefore, compared with Mandya et al. (2020), our A-GCN encodes the dependency connections and dependency types in a more intensive manner and thus can better leverage them to guide the process of predicting the relations between the given entities.

5 Analyses

5.1 The Effect of A-GCN

Dependency information is supposed to be beneficial for RE because it contains long-distance word-word relations, which could be extremely useful when the given two entities are far away from each other in the input sentence. To explore the effect of A-GCN in capturing such long-distance word-word relations to help with RE, we split the test instances into different groups according to their entities' distances (i.e., the number of words between the two entities) and run models on these groups to test their performance. Figure 4 shows the performance of our best performing A-GCN model with BERT-large (ID: 11 in Table 2) and its corresponding standard GCN and BERT-large baselines on the three groups of test instances from the test set of SemEval, where the category name indicates the range of the entity distance.[17] It is observed that, A-GCN outperforms the two baselines on all groups of test instances and the improvement becomes larger when the entity distance increases.

[17]For example, a test sentence whose distance in between two entities is 7 will fall into the group $(5, 10]$.

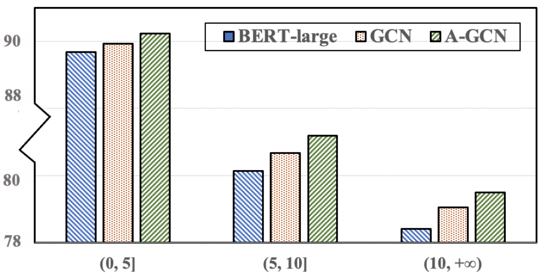


Figure 4: Performance (F1 scores) of different models (i.e., BERT-only, two layers of standard GCN, and two layers of A-GCN) with the BERT-large encoder on three groups of test instances from SemEval, where each group is generated based on the distance (i.e., number of words) between two entities in an instance.

This observation confirms that our approach is able to leverage dependency information and capture long-distance word-word relations to improve RE.

5.2 The Effect of Graph Construction

In the main experiments, we try A-GCN with the graph built upon the combined local and global connections ("L + G"). To explore the effect of the local connections and the global connections for A-GCN, we run our approach using two A-GCN layers with the graph constructed by local connections ("L") or global connections ("G") alone. Table 4 presents the experimental results (F1 scores) of different models with BERT-base and BERT-large encoders, where the results from BERT-only baselines, A-GCN (L + G), and A-GCN (Full) are also copied from Table 2 for reference. Compared to A-GCN (L + G), models with the graph constructed by either local connections (i.e., A-GCN (L)) or global connections (i.e., A-GCN (G)) achieve lower performance, which complies with our intuition because both groups of connections contain important contextual features for RE. Interestingly, it is found that A-GCN (L) outperforms A-GCN (G) with both BERT-base and BERT-large encoders. A possible explanation could be the following. There are overlaps between local and global connections (e.g., the connection between "*range*" and "*restrictions*" in Figure 3). Therefore, A-GCN (L) can not only leverage the contextual information associated with the entities themselves, but is also partially[18] benefited from the overlapping connections on the SDP between the two entities, which leads A-GCN (L) to achieve a higher performance than A-GCN (G).

[18]When there is only one word on the shortest dependency path between two entities, all global connections are included in local ones, e.g., "*defamation*" and "*bishop*" in Figure 2.

ID	Models	ACE2005	SemEval
1	BERT-Base	75.31	87.87
2	+ A-GCN (L)	76.92	88.89
3	+ A-GCN (G)	76.72	88.89
4	+ A-GCN (L + G)	**77.30**	**89.16**
5	+ A-GCN (Full)	77.25	88.70
6	BERT-Large	76.79	89.02
7	+ A-GCN (L)	78.61	89.70
8	+ A-GCN (G)	78.40	89.38
9	+ A-GCN (L + G)	**79.05**	**89.85**
10	+ A-GCN (Full)	78.91	89.70

Table 4: Performance of our models with two A-GCN layers using the graphs built upon (1) only local connections (L), (2) only global connections (G), (3) the combination of local and global connections (G + L) , and (4) full dependency graph (Full). The performance of BERT-only baseline is also reported for reference.

	Att.	Type	ACE2005	SemEval
BERT-base	Baseline		75.31	87.87
	√	√	**77.30**	**89.16**
	×	√	77.00	88.07
	√	×	76.27	88.50
	GCN		76.11	88.62
BERT-large	Baseline		76.79	89.02
	√	√	**79.05**	**89.85**
	×	√	78.92	89.26
	√	×	78.22	89.37
	GCN		77.92	89.13

Table 5: The ablation study on the attention mechanism (Att.) and dependency types (Type) in our best model, i.e., two layers of A-GCN (L + G). "$\surd$" and "$\times$" stand for that whether a module is used. The F1 scores of BERT-only and the standard two layers of GCN (L + G) are also reported for references.

5.3 Ablation Study

Compared with the standard GCN, A-GCN enhances it from two aspects: (1) using an attention mechanism to weigh different dependency connections and (2) introducing dependency types to the process to encode more detailed dependency information. To better investigate the effect of each individual enhancement (i.e., the attention mechanism or the dependency type information), we conduct an ablation study on our best model, i.e., two layers of A-GCN (L + G) with BERT-base and BERT-large encoder. Table 5 reports the experimental results of different models, where the performance of BERT-only baseline and the standard GCN baseline (i.e., the one uses neither the attention mechanism nor dependency types) are also reported for reference. The results clearly indicate that, the ablation of either enhancement (i.e., the attention mechanism or the dependency type information) could result in worse results (compared with full A-GCN). Between the two enhancements, the ablation of the attention mechanism hurts A-GCN more, which indicates the ability of distinguishing important connections and leveraging them accordingly plays a more important role in RE.

5.4 Case Study

To explore in detail that how A-GCN leverages dependency connections and types to improve RE, we conduct a case study with our A-GCN models with different dependency graphs (i.e., two layers of A-GCN (Full) and A-GCN (L + G) with BERT-large encoder) on an example sentence "*A central vacuum is a vacuum motor and filtration system built inside a canister.*". Figure 5 shows the sentence where both the two models correctly predict the relation between "*motor*" (E_1) and "*canister*" (E_2) (highlighted in the red color) to be "*Content-Container*", whereas the baseline GCN (Full) and GCN (L + G) models fail to do so. We also visualize the attention weights assigned to different dependency connections extracted from the last A-GCN layer, with darker and thicker lines referring to higher weights. In this example, for A-GCN (Full), we observe that the connection between "*built*" and "*canister*" along SDP and the connection between "*inside*" and "*canister*" receive the highest weights, where this is valid because the dependency type, i.e., *obl* (oblique nominal), associated with the connection (between "*built*" and "*canister*") reveals that "*canister*" could be the position where the action (i.e., *build*) takes place, and is further confirmed by another dependency connection and type (i.e., *case*) between "*inside*" and "*canister*". Therefore, it is proved that our model learn from the contextual information carried by such important connections and results in correct RE prediction. Similarly, A-GCN (L + G) also correctly perform RE on this case by highlighting the same dependency connections as those from the A-GCN (Full) with much higher weights (because many dependency connections are filtered out).

6 Related Work

Recently, neural networks with integrating external knowledge or resources play important roles in RE because of their superiority in better capturing contextual information (Shen and Huang,

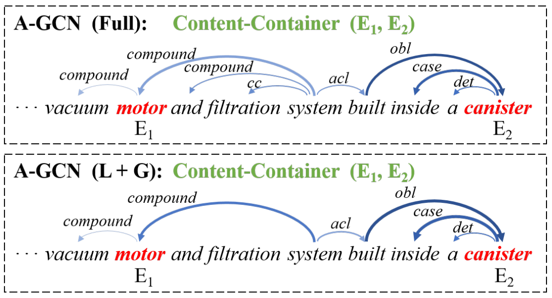

A-GCN (Full): Content-Container (E1, E2)
compound
compound
compound
cc
acl
obl
case
det
··· vacuum motor and filtration system built inside a canister
E1
E2
A-GCN (L + G): Content-Container (E1, E2)
compound
compound
acl
obl
case
det
··· vacuum motor and filtration system built inside a canister
E1
E2

Figure 5: Visualizations of weights assigned to different dependency connections of A-GCN (Full) and A-GCN (L + G) for an example input, where darker and thicker lines refer to connections with higher weights.

2016; Soares et al., 2019). Particularly, as one kind of such knowledge, dependency parses show their effectiveness in supporting RE for its ability in capturing long-distance word relations (Zhang et al., 2018; Guo et al., 2019). However, intensively leveraging dependency information could introduce confusions to RE (Xu et al., 2016b; Yu et al., 2020) so that necessary pruning is required to alleviate this problem. E.g., Xu et al. (2015) proposed to use the connections along the shortest dependency path between the two entities and apply LSTM to model them; Miwa and Bansal (2016) proposed to prune the original dependency tree into the lowest common ancestor subtree. However, these pruning strategies are either too aggressive or modest, so that the resulted graph might lose some important contexts or filled with more noise. Zhang et al. (2018) adopted GCN to model the dependencies and proposed a trade-off pruning strategy in between Xu et al. (2015) and Miwa and Bansal (2016). Besides, there are other graph-based models for RE that utilize layers of multi-head attentions (Guo et al., 2019), dynamic pruning (Yu et al., 2020), and additional attention layers (Mandya et al., 2020) to encode dependency trees. Compared with the aforementioned methods, especially the graph-based ones, our approach offers an alternative to enhance RE with A-GCN by using attention mechanism and dependency type, which are effective and efficient improvement to standard GCN without requiring complicated model design.

7 Conclusion

In this paper, we propose A-GCN to leverage dependency information for relation extraction, where an attention mechanism is applied to dependency connections to applying weighting on both connections and types so as to better distinguish the important dependency information and leverage them accordingly. In doing so, A-GCN is able to dynamically learn from different dependency connections so that less-informative dependencies are smartly pruned. Experimental results and analyses on two English benchmark datasets for relation extraction demonstrate the effectiveness of our approach, especially for entities with long word-sequence distances, where state-of-the-art performance is obtained on both datasets.

Acknowledgements

This work is supported by Chinese Key-Area Research and Development Program of Guangdong Province (2020B0101350001) and NSFC under the project "The Essential Algorithms and Technologies for Standardized Analytics of Clinical Texts" (12026610). This work is also partially supported by Shenzhen Institute of Artificial Intelligence and Robotics for Society under the project "Automatic Knowledge Enhanced Natural Language Understanding and Its Applications" (AC01202101001). We also thank Mr. Peilin Zhou for providing the first version of the model architecture figure.

References

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based Model for Arabic Language Understanding. *arXiv preprint arXiv:2003.00104*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

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. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*.

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.

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. 2020. ZEN: Pre-training Chinese Text Encoder Enhanced by N-gram Representations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4729–4740.

Bayu Distiawan, 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.

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.

Valerie Hajdik, Jan Buys, Michael Wayne Goodman, and Emily M. Bender. 2019. Neural Text Generation from Rich Semantic Representations. 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 2259–2266, Minneapolis, Minnesota.

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.

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.

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. 2021. 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*.

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.

Yatian Shen and Xuanjing Huang. 2016. Attention-based Convolutional Neural Network for Semantic Relation Extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2526–2536, Osaka, Japan.

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.

Yan Song, Prescott Klassen, Fei Xia, and Chunyu Kit. 2012. Entropy-based Training Data Selection for Domain Adaptation. In *Proceedings of COLING 2012: Posters*, pages 1191–1200.

Yan Song, Chia-Jung Lee, and Fei Xia. 2017. Learning Word Representations with Regularization from Prior Knowledge. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 143–152.

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. 2018. 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, Yuanhe Tian, Nan Wang, and Fei Xia. 2020. Summarizing Medical Conversations via Identifying Important Utterances. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 717–729.

Yan Song and Fei Xia. 2013. A Common Case of Jekyll and Hyde: The Synergistic Effect of Using Divided Source Training Data for Feature Augmentation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 623–631.

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.

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 5683–5692.

Yuanhe Tian, Guimin Chen, and Yan Song. 2021a. 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, Online.

Yuanhe Tian, Guimin Chen, and Yan Song. 2021b. 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, Online.

Yuanhe Tian, Wang Shen, Yan Song, Fei Xia, Min He, and Kenli Li. 2020a. Improving Biomedical Named Entity Recognition with Syntactic Information. *BMC Bioinformatics*, 21:1471–2105.

Yuanhe Tian, Yan Song, and Fei Xia. 2020b. 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. 2020c. 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. 2020d. Improving Constituency Parsing with Span Attention. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1691–1703.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph Attention Networks. *arXiv preprint arXiv:1710.10903*.

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.

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.

Shanchan Wu and Yifan He. 2019. Enriching Pretrained Language Model with Entity Information for Relation Classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2361–2364.

Zhaofeng Wu, Yan Song, Sicong Huang, Yuanhe Tian, and Fei Xia. 2019. WTMED at MEDIQA 2019: A Hybrid Approach to Biomedical Natural Language Inference. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 415–426, Florence, Italy.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016a. 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.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016b. 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)*.

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.

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.

Bowen Yu, Mengge Xue, 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).

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*.

Hongming Zhang, Yan Song, and Yangqiu Song. 2019. Incorporating Context and External Knowledge for Pronoun Coreference Resolution. 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 872–881.

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. Hyper-parameter Settings

Table 6 reports the hyper-parameters tested in training our models. We test all combinations of them for each model and use the one achieving the highest F1 score in our final experiments. The best hyper-parameter setting is highlighted in boldface.

Hyper-parameters	**Values**
Learning Rate	$5e-6, 1e-5, 2e-5, \mathbf{3e-5}$
Warmup Rate	$\mathbf{0.06}, 0.1$
Dropout Rate	$\mathbf{0.1}$
Batch Size	$16, \mathbf{32}, 64, 128$

Table 6: The hyper-parameters tested in tuning our models. The best ones used in our final experiments are highlighted in boldface.

Appendix B. Model Size and Running Speed

Table 7 reports the number of trainable parameters and the inference speed (sentences per second) of the baseline (i.e., BERT, BERT + GAT and BERT + GCN) and our models (i.e., BERT + A-GCN) on ACE2005 and SemEval datasets. All models are performed on an NVIDIA Tesla V100 GPU.

Appendix C. Experimental Results on the Development Set

Table 8 reports the F1 scores of different models on the development set of ACE2005.[19]

Appendix D. Mean and Deviation of the Results

In the experiments, we test models with different configurations. For each model, we train it with the best hyper-parameter setting using five different random seeds. We report the mean (μ) and standard deviation (σ) of the F1 scores on the test set of ACE2005 and SemEval in Table 9.

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

Models	**ACE2005**		**SemEval**	
	Para.	Speed	Para.	Speed
BERT-base	109M	27.7	109M	54.7
+ GAT (Full)	110M	26.2	110M	51.8
+ GAT (L + G)	110M	26.2	110M	51.8
+ 1 GCN layer (Full)	110M	26.4	110M	52.2
+ 1 A-GCN layer (Full)	110M	25.1	110M	50.4
+ 1 GCN layer (L + G)	110M	26.4	110M	52.2
+ 1 A-GCN layer (L + G)	110M	25.1	110M	50.4
+ 2 GCN layers (Full)	111M	24.8	111M	49.9
+ 2 A-GCN layers (Full)	111M	24.1	111M	48.7
+ 2 GCN layers (L + G)	111M	24.8	111M	49.9
+ 2 A-GCN layers (L + G)	111M	24.1	111M	48.7
+ 3 GCN layers (Full)	112M	23.1	112M	47.9
+ 3 A-GCN layers (Full)	112M	23.0	112M	47.2
+ 3 GCN layers (L + G)	112M	23.1	112M	47.9
+ 3 A-GCN layers (L + G)	112M	23.0	112M	47.2

(a) BERT-base

Models	**ACE2005**		**SemEval**	
	Para.	Speed	Para.	Speed
BERT-large	335M	8.9	335M	17.1
+ GAT (Full)	337M	8.4	337M	16.7
+ GAT (L + G)	337M	8.4	337M	16.7
+ 1 GCN layer (Full)	337M	8.6	337M	16.9
+ 1 A-GCN layer (Full)	337M	8.1	337M	16.6
+ 1 GCN layer (L + G)	337M	8.6	337M	16.9
+ 1 A-GCN layer (L + G)	337M	8.1	337M	16.6
+ 2 GCN layers (Full)	338M	8.0	338M	16.3
+ 2 A-GCN layers (Full)	338M	7.8	338M	16.1
+ 2 GCN layers (L + G)	338M	8.0	338M	16.3
+ 2 A-GCN layers (L + G)	338M	7.8	338M	16.1
+ 3 GCN layers (Full)	339M	7.4	339M	15.8
+ 3 A-GCN layers (Full)	339M	7.2	339M	15.5
+ 3 GCN layers (L + G)	339M	7.4	339M	15.8
+ 3 A-GCN layers (L + G)	339M	7.2	339M	15.5

(b) BERT-large

Table 7: Numbers of trainable parameters (Para.) in different models and the inference speed (sentences per second) of these models on the test sets of both datasets.

Models	**BERT-base**	**BERT-Large**
Baseline	75.03	76.11
GAT (Full)	75.33	76.87
GAT (L + G)	75.31	76.93
+ 1 GCN layer (Full)	74.97	76.13
+ 1 A-GCN layer (Full)	76.49	77.33
+ 1 GCN layer (L + G)	75.80	77.19
+ 1 A-GCN layer (L + G)	76.00	77.49
+ 2 GCN layers (Full)	75.36	77.35
+ 2 A-GCN layers (Full)	76.65	77.55
+ 2 GCN layers (L + G)	76.59	77.48
+ 2 A-GCN layers (L + G)	**76.90**	**77.82**
+ 3 GCN layers (Full)	75.61	77.33
+ 3 A-GCN layers (Full)	76.45	77.54
+ 3 GCN layers (L + G)	76.48	77.36
+ 3 A-GCN layers (L + G)	76.58	77.65

Table 8: F1 scores of our A-GCN models and the baselines (i.e., BERT-only, standard GAT, and standard GCN) under different settings with BERT-base and BERT-large on the development set of ACE2005. All graph-based models (i.e., GAT, GCN, and A-GCN) are tested with two settings: the first is using the full graph (FULL) with all dependency connections involved and the second is using the combination of local and global connections (L + G). We also run GCN and A-GCN with different numbers of layers (i.e., 1 to 3 layers) for fair comparisons.

Models	ACE2005		SemEval	
	μ	σ	μ	σ
BERT-base	75.22	0.31	87.39	0.26
+ GAT (Full)	75.87	0.23	88.16	0.44
+ GAT (L + G)	75.47	0.27	88.15	0.28
+ 1 GCN layer (Full)	74.51	0.13	87.34	0.29
+ 1 A-GCN layer (Full)	74.39	0.21	88.02	0.30
+ 1 GCN layer (L + G)	75.28	0.23	88.43	0.17
+ 1 A-GCN layer (L + G)	76.70	0.37	88.69	0.28
+ 2 GCN layers (Full)	74.73	0.24	88.13	0.31
+ 2 A-GCN layers (Full)	76.95	0.21	88.35	0.34
+ 2 GCN layers (L + G)	75.60	0.42	88.30	0.23
+ 2 A-GCN layers (L + G)	**77.06**	0.13	**88.81**	0.28
+ 3 GCN layers (Full)	75.37	0.15	88.26	0.21
+ 3 A-GCN layers (Full)	75.94	0.28	88.29	0.26
+ 3 GCN layers (L + G)	76.48	0.38	88.10	0.16
+ 3 A-GCN layers (L + G)	75.87	0.45	88.46	0.25

(a) BERT-base

Models	ACE2005		SemEval	
	μ	σ	μ	σ
BERT-large	76.55	0.17	88.63	0.26
+ GAT (Full)	77.96	0.18	89.10	0.21
+ GAT (L + G)	78.33	0.38	89.13	0.31
+ 1 GCN layer (Full)	77.30	0.28	88.52	0.31
+ 1 A-GCN layer (Full)	78.15	0.37	89.05	0.49
+ 1 GCN layer (L + G)	76.98	0.49	88.80	0.28
+ 1 A-GCN layer (L + G)	78.04	0.32	89.32	0.22
+ 2 GCN layers (Full)	78.56	0.41	89.16	0.26
+ 2 A-GCN layers (Full)	78.68	0.22	89.34	0.33
+ 2 GCN layers (L + G)	78.40	0.33	89.22	0.17
+ 2 A-GCN layers (L + G)	**78.83**	0.21	**89.41**	0.44
+ 3 GCN layers (Full)	77.58	0.32	89.14	0.36
+ 3 A-GCN layers (Full)	78.03	0.32	89.16	0.17
+ 3 GCN layers (L + G)	78.64	0.27	88.93	0.26
+ 3 A-GCN layers (L + G)	78.55	0.45	89.20	0.33

(b) BERT-large

Table 9: The mean μ and standard deviation σ of F1 scores of our A-GCN model and baselines on the test set of ACE2005 and SemEval for relation extraction.