Self-Attention Graph Residual Convolutional Network for Event Detection with dependency relations

Haozhe Liu and **Ning Xu*** and **Anan Liu***
Schoo of Electrical and Information Engineering, Tianjin University. Tianjin, China
liuhz0305@gmail.com
ningxu@tju.edu.cn
anan0422@gmail.com

Abstract

Event detection (ED) task aims to classify events by identifying key event trigger words embedded in a piece of text. Previous research have proved the validity of fusing syntactic dependency relations into Graph Convolutional Networks(GCN). While existing GCN-based methods explore latent node-to-node dependency relations according to a stationary adjacency tensor, an attention-based dynamic tensor, which can pay much attention to the key node like event trigger or its neighboring nodes, has not been developed. Simultaneously, suffering from the phenomenon of graph information vanishing caused by the symmetric adjacency tensor, existing GCN models can not achieve higher overall performance. In this paper, we propose a novel model Self-Attention Graph Residual Convolution Networks (SA-GRCN) to mine node-to-node latent dependency relations via self-attention mechanism and introduce Graph Residual Network (GResNet) to solve graph information vanishing problem. Specifically, a self-attention module is constructed to generate an attention tensor, representing the dependency attention scores of all words in the sentence. Furthermore, a graph residual term is added to the baseline SA-GCN to construct a GResNet. Considering the syntactically connection of the network input, we initialize the raw adjacency tensor without processed by the self-attention module as the residual term. We conduct experiments on the ACE2005 dataset and the results show significant improvement over competitive baseline methods.

1 Introduction

Event Detection(ED) is an important information extraction task that aims to match patterns of events in a context.The event patterns and the event contexts define event types.(Xiang and Wang, 2019)(Mellin and Berndtsson, 2009) Generally, the event type of each event context is labeled by a trigger word or phrase, called “event trigger”. However, when performing event detection from an sentence-level perspective, one of the most common scenarios is that there will be multiple event triggers in the same sentence, which means that it is necessary to recognize all the event triggers and classify them into specific event types. Taking figure 1 as an example, ED is supposed to recognize the event trigger “death” and “wounded” and classify them to the event type “Die” and “Injure”.

However, existing GCN-based ED methods(Liu et al., 2018) updates the graph by an adjacency tensor which results from the syntactic analysis. Such a graph structure only pay attention to the directly connected nodes. As shown in Figure 1, the dependency labels “nsubj” (nominal subject) and “dobj”(direct object) show that “Center”and “deaths” are directly connected to the root node “recorded” respectively. However, for the event type “Injure”, event trigger “wounded” can not be recognized even though it’s connected to the “dobj” node “deaths” with “nmod” (noun compound modifier) dependency. Such an observation indicates that for multiple ED tasks, some event triggers may be ignored if they are indirectly connected to the root node. Therefore, to perform multiple ED tasks on a single sentence, it’s infeasible to rely solely on the dependency labels obtained by syntactic analysis. It’s necessary to pay attention to the event triggers which are indirectly connected to the root node.

In addition, even for the nodes which are indirectly connected with the dependency labels “nsubj” and “dobj” may assist detecting the single event trigger. Specifically, taking Figure 1 as an example: the "nsubj" dependency label associates "Center" with "death", but the “appos” (apposition modifier) of "Center", "hospital", has a closer dependency relations with "death". As the entity labeled by the ACE2005, both "Center" and "hospi-

*Corresponding authors.

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 302–311
November 7–11, 2021.

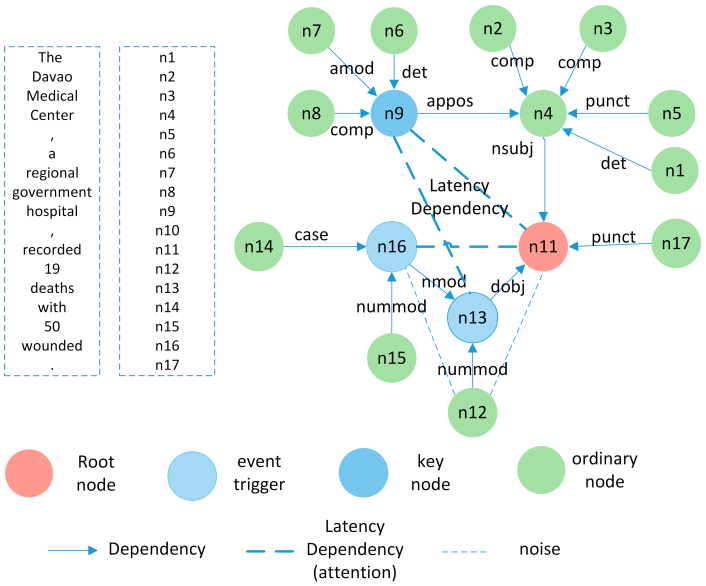

Figure 1: The graph structure constrcuted by the dependency relation labels.

tal" are also classified as "ORG-Medical-Science". In this case, modifier node like “hospital” should be viewed as a key node to discover latent dependency relations, in other words, node like "hospital" should be giving a higher attention weights when updating the graph. Although EE-GCN (Cui et al., 2020b) proposed to explore latent dependency relations by aggregating information from neighbors of each node through specific edge, it will also edge-enahance noisy dependency relations. Nosiy nodes like “punct”, “det”connected with key nodes will interfere the event detection and dependency relations like “nmod” or “nummod” will be wrongly constituted between irrelevant neighboring nodes. It cannot efficiently locate the event trigger nodes in the whole graph. Therefore, it’s reasonable to apply the attention mechanism for more in-depth tap the latent node-to-node dependency relations.

Further more, as mentioned in the GResNet(Zhang and Meng, 2019), the suspended animation limit problem of the existing GNNs caused the information vanishing phenomenon, which means the model will respond nothing to the training data and become unlearnable when the model depth reaches such an limit. Considering the syntactic connection between the word sequences, we used the encoded GCN input as the graph residual term to solve this problem.

In this paper, we propose a novel architecture named Self-Attention Graph Residual Convolutional Networks(SA-GRCN), which making full use of syntactic dependency labels to generate the graph and simultaneously giving different attention weights to the nodes on the dependency graph. Resembling the existing methods, we transforms a sentence to a graph by viewing words and dependency labels as nodes and typed edges. An asymmetric adjacency tensor is initialized to represent the graph, considering the directionality of the dependency labels. To explore node-to-node latent dependency relations, a self-attention module is constructed to update the self-attention tensor of the whole graph. Node-aware edge update module and edge-aware node update module are used to update the network iterativly, such mutual update process make the latent dependency relations expressed in the node representations can be effectively mined and injected to the attention tensor. Via a dynamic updated self-attention tensor and a initialized adjacency tensor, our architecture can better capture the interrelations between candidate trigger words and related entities.

Our contributions are summarized as follows:

We propose the novel Self-Attention Graph Residual Convolutional Network, introducing the attention mechanism into GCN, which simultaneously integratse syntactic structure and latent dependency relations to improve the performance of event detection based on existing GCN methods. What’s more, GResNet is introduces to deal with the phenomenon of graph information vanishing. To our best our knowledge, this is the first time to

apply self-attention mechanism to the ED task.

• Different from the conventional binary adjacency matrix, we propose two asymmetric adjacency tensors for edge representation of the graph structure and an attention tensor which calculates attention scores of all the words in a sentence to explore the latent dependency relatiions.

• Experiments conducted on the ACE2005 * benchmark show that SA-GRCN achieves the state-of-art overall performance, especilly on the excellent performance in precision.

2 Related works

In earlier studies, researchers perform event detection task based on statistic features, modeling the statistic distributions to cluster the event. However, all of these manually defined features, which are highly dependent on a specific data set, demand high for the preprocessing module. Consequently, the robustness of all these models are poor.

In recent years, with the rapid development of machine learning, many ED-oriented neural networks have been proposed (Chen et al., 2015)(Nguyen et al., 2016)(Feng et al., 2016). The existing approaches can be categorized into two classes: The first class is to improve ED through special learning techniques including adversarial training (Hong et al., 2018), knowledge distillation (Lu et al., 2019) and model pre-training (Yang et al., 2019). The second class is to improve ED by introducing extra resource, such as argument information (Liu et al., 2017) , document information (Duan et al., 2017)(Zhao et al., 2018)(Chen et al., 2017), knowledge base (Liu et al., 2016) and syntactic dependency labels (Tang et al., 2020) (Cui et al., 2020b).

Dependcy labels can be viewed as a kind of syntactic information (Cui et al., 2020a) (Lai et al., 2020)(Schlichtkrull et al., 2018), which is essential for the GCN-based ED task. (Cui et al., 2020b) exploited typed-dependency labels to integrated dependency tree into GCN models. Compare with the binary adjacency tensor of Vanilla GCN, (Cui et al., 2020b) construct a symmetric adjacency tensor via the dependency trees to denote the graph structure. (Cui et al., 2020b) also proposed edge-enhanced to explore the latency dependency relations, but it also brings a lot of useless noisy relations. Although (Yan et al., 2019) tried to apply attention mechanism into ED task, it suffers from the information vanishing of node-to-node features. Here, we improved the performance based on the above methods. How to effectively leverage the typed dependency information still remains a challenge in this task.

*https://catalog.ldc.upenn.edu/LDC2006T06

3 Methods

In this section, we will introduce our Self-Attention Graph Residual Convolutional Network (SA-GRCN) architecture to explore potential event triggers including the Embedding method, RNN Encode module,Self-Attention Collect module and GResNet. Furthermore, we introduce a residual term to our novel architecture to solve the information vanishing phenomenon caused by the symmetric adjacency tensor.

3.1 Self-Attention Graph Residual Convolutional Network

Edge-Enhanced GCN(Cui et al., 2020b) (EE-GCN) first incorporates typed dependency label information into the feature aggregation process to obtain better representations. Specifically, EE-GCN constructs an adjacency tensor $\mathbf{E} \in \mathbb{R}_{n\times n\times p}$ to describe the graph structure instead of the binary adjacency matrix used in the vanilla GCN, where $\mathbf{E}_{i,j,:} \in \mathbb{R}_p$ is the p-dimensional relation representation between node i and node j, and p can also be understood as the number of channels in the adjacency tensor. Formally, $\mathbf{E}$ is initialized according to the dependency tree, if a dependency edge exists between w_i and w_j and the dependency label is r, then $\mathbf{E}_{i,j,:}$ is initialized to the embedding of r obtained from a trainable embedding lookup table, otherwise initialize $\mathbf{E}_{i,j,:}$ with a p-dimensional all-zero vector. Following previous works (Zhang et al., 2018) (Guo et al., 2019), $\mathbf{E}$ is initialized based on an undirectional graph, which means that $\mathbf{E}_{i,j,:}$ and $\mathbf{E}_{j,i,:}$ are initialized as the same embedding.

Our proposed SA-GRCN is an extension of the GCN mentioned above, making full use of syntactic dependency label information and simultaneously giving different attention weights to the nodes on the dependency graph to locate event triggers. Specially, apart from the edge representation tensor $\mathbf{E}$ and node representation tensor $\mathbf{H}$, SA-GRCN constructs another input tensor $\mathbf{D} \in \mathbb{R}_{n\times dep_{size}\times p}$ to describe the node-dependency features of the network, denoting the

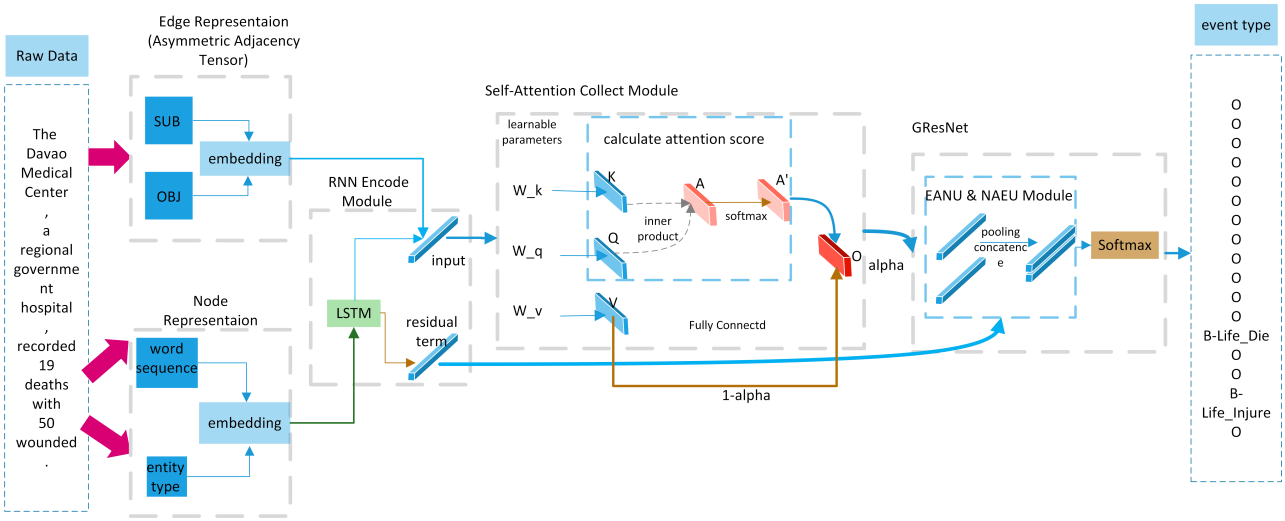

Figure 2: The framework of Self-Attention Graph Residual Convolutioinal Network.

encoded sentence-level data processed by the word embdding, entity type embedding and dependency embedding. This input tensor can initialze each word in the sentence as a dep_{size} dimension vectors, corresponding to the number of dependency labels.

In order to fully leverage all of the input tensor and effectively mine latent relation information beyond the dependency labels, three modules are implemented at each layer of SAEE-GCN to update the node representations (H) and edge representations (E) mutually through information aggregation:

$$\mathbf{H}^l, \mathbf{E}^l = SA - GRCN(\mathbf{H}^{l-1}, \mathbf{D}^{l-1}, \mathbf{E}^{l-1}).$$

For the Vanilla GCN or the EE-GCN, the input tensor can be devided into two part. The first part is the source data which processed by the word embedding and entity type embedding. The second one is the adjaceny tensor processed by the dependency embedding. Similar to the entity type embedding, adjacency tensor is a symmetric tensor which can represent the node-node dependency relations. Each dependency labels is assign to a real-value according to the embedding table. However, such a symmetric tensor only contains the topological structure of a whole sentence, ignoring directionality of the subject and the object. Inspired by (Yang et al., 2018), we mapped the attention relations in the scene graph to the event detection task. In fact, each group of dependency relation in the word sequences is directional, which can be regarded as the subject node pointing to the object node. Therefore, each word can be either subject node or object node. Therefore, we proposed to utilize two asymmetric tensors **SUB** and **OBJ** to generate edge representation tensor **E**, so as to denote the relationship bewteen subject word and object word and dependency relations respectively.

3.1.1 Embedding and RNN Encode Module

As mentioned above, we performed a sentence-level event detection, considering that there will be one or more event triggers in the same sentence. Similar to the previous work, we defined $S = \{w_1, w_2, \ldots, w_n\}$ to denote an n-word sentence.

• Word embedding w_i: it captures the meaningful semantic regularity of word. Following previous works (Chen et al., 2018)(Yan et al., 2019), we use the word embedding pre-trained by Skip-gram on the NYT Corpus.

• Entity type embedding e_i: entities in the sentence are annotated with BIO schema and we map each entity type label to a real-valued embedding by looking up an embedding table.

• Dependency embedding: The results of the semantic analysis tools contains the subject and the object of a word pair, which are connected with a typed dependency label, directing from the subject node to the object node. It initializes the adjacency tensor of the GCN according to the semantic analysis results of the target sentence. Since each word can be viewed as a subject node in one dependency relation and a object node in the other. In this paper, we realzie the edge representation via two asym-

metric tensors **SUB** and **OBJ**. The hidden size of the dependency embedding is the number of the typed dependency labels.

Further more, to apply the attention mechanism to the GCN-based methods, we proposed to match the source data of the GCN input to the relationship between word and dependency relations, which means that the dimension of the GCN input becomes $n \times dep_{size}$ instead of the conventional $n \times dim$. Such an initialized node features can be appropriate for the self-attention mechanism to explore latent semantic dependency relations between word sequences.

Thus, the input embedding of w_i can be defined as $x_i = [w_i; e_i] \in \mathbb{R}^{d_w+d_e}$, where d_w and d_e denote the dimension of word embedding and entity type embedding respectively. Then, a BiLSTM layer is adopted to capture the contextual information for each word. For simplicity, we denote the contextualized word representations as $S_1, S_2 = [h_1, \ldots, h_n]$, where $S_1 \in \mathbb{R}_{n \times dep_{size}}$ and $S_2 \in \mathbb{R}_{n \times d}$ are used as initial node features in GCN.

3.1.2 Self-Attention Collect Module

The self-attention opertion can be formulated as:

$$K = W^k \times I, \quad Q = W^q \times I, \quad A = K^T \times Q.$$

where the input tensor I is $\mathbf{D}$ encoded by the initialized asymmetric tensor **SUB** or **OBJ**.

$$I_S = \mathbf{SUB} \times \mathbf{D}, \quad I_O = \mathbf{OBJ} \times \mathbf{D}.$$

K and Q are aimed at utilizing dot product operation to calculate the self-attention score. And then, extract information based on attention scores to get the attention outputs. The attention tensor A denotes the relevant dependency relations between word sequences.

$$V = W^v \times I, \quad O = V \times A.$$

In such a self-attention mechanism, three weight parameters W^k, W^q, W^v can be learned during the training process. The dimsension of these threee matrices is $dep_{size} \times d$, which means the hidden size of these learnable parameters represents the number of dependency labels. Specifically, for the subject and object input tensor, the results of the self-attention module are $\mathbf{O}_{sd}$ and $\mathbf{O}_{od}$. The attention outputs for node i can be formulated as:

$$\mathbf{h}_i^{attention} = \mathbf{h}_i^{initial} + \mathbf{O}_{sd} + \mathbf{O}_{od}$$

In this way, the $n \times d$ dimension attention outputs can explore the latent node-dependency relations during training process.

3.1.3 GResNet

(Zhang and Meng, 2019) provided an analysis about the suspended animation problem of the spectral graph convolutional operator used in GCN to interpret the information vanishing phenomenon.

For Vanilla GCN model, the corresponding node representation updating equations can be formulated as:

$$\begin{cases} \mathbf{H}^{(0)} = \mathbf{X}, \\ \mathbf{H}^{(k)} = ReLU(\hat{\mathbf{A}}\mathbf{H}^{(k-1)}\mathbf{W}^{(k-1)}), \\ \hat{\mathbf{Y}} = softmax(\hat{\mathbf{A}}\mathbf{H}^{(K-1)}\mathbf{W}^{(K-1)}). \end{cases}$$

where K denotes the depth of the GCN model and $\forall k \in \{1, 2, \ldots, K-1\}$. The spectral graph convolutional operator defined above can be divided into the following two sequential steps :

$$\begin{cases} MC\ Layer : \mathbf{T}^{(k)} = \hat{\mathbf{A}}\mathbf{H}^{(k-1)}, \\ FC\ Layer : \mathbf{H}^{(k)} = ReLU(\mathbf{T}^{(k)}\mathbf{W}^{(k-1)}). \end{cases}$$

where the first term on the right-hand-side defines a 1-step Markov chain (MC or a random walk) based on the graph and the second term is a fully-connected (FC) layer parameterized by variable $\mathbf{W}^{(k-1)}$.

Considering that the variables $\mathbf{W}^{(k-1)}$ for the vector dimension adjustment are shared among all the nodes, given two nodes with identical representations, the mapping defined by fully-connected layers becomes identity mapping. Meanwhile, the Markov chain layers may converge with k layers iff $\mathbf{T}^{(k)} = \mathbf{T}^{(k-1)}$, i.e., the representations before and after the updating are identical (or very close), which is highly dependent on the input network structure, i.e., matrix $\hat{\mathbf{A}}$, actually.

We can derive similar results for the multiple Markov chain layers in the EE-GCN model based on the nodes' feature inputs, which will reduce the learned nodes' representations to the stationary representation tensor. For EE-GCN, the initialized adjacency tensor $\mathbf{E}$ can describe the graph structure. However, according to the analysis listed above, the stationary distribution vector of $\mathbf{E}$ will finally becomes a uniform distribution over nodes during the updating process, causing the information vanishing phenomenon. Therefore, to basically

solve such a information vanishing problem, we generate two asymmetric tensors **SUB** and **OBJ** to describe the edge representation instead of a conventional symmetric adjacency tensor.

A more robust method to solve information vanishing is to introduce residual networks. Residual learning initially introduced in (He et al., 2015) divides the objectActive mapping into two parts: the inputs and the residual function. For instance, let $H(x)$ be the objective mapping which projects input x to the desired domain. The ResNet introduced in (He et al., 2015) divides $H(x)$ as $F(x) + R(x)$ (where $R(x) = x$ is used in (He et al., 2015)). This reformulation is motivated by the counterintuitive phenomena about the degradation problem observed on the deep CNN. Different from the learning settings of CNN, where the data instances are assumed to be independent, the nodes inside the input network studied in GCN are closely correlated. Viewed in such a perspective, new residual learning mechanism should be introduced for GCN specifically.

$$\begin{cases} \mathbf{H}^{(0)} = \mathbf{X}, \\ \mathbf{H}^{(k)} = ReLU(\alpha\mathbf{H}^{(k-1)}\mathbf{W}^{(k-1)} + (1-\alpha)\mathbf{R}, \\ \hat{\mathbf{Y}} = softmax(\alpha\mathbf{H}^{(K-1)}\mathbf{W}^{(K-1)} + (1-\alpha)\mathbf{R}. \end{cases}$$

where the residual term $\mathbf{R}$ is the initialized input tensor $\mathbf{H}$ encoded by the word embedding, entity type embedding and dependency embedding, representing node features of the whole graph. Such a skip can effectively solve the information vanishing problem during the training process. α is a hyper parameter which can balance the self-attention layer output and the initialized input. Our experimental results show that the introduction of GResNet can indeed solve the problem of information vanishing to a certain extent.

3.1.4 Edge-aware node update and Node-aware edge update

The Edge update method was first proposed in EE-GCN, but compared to the core algorithm of Edge-Enhanced operation is based on a stationary adjacency tensor, Self-Attention method can dynamically update according to attention tensor.

Considering that the attention outputs can represent node-dependency features, with words in sentence interpreted as nodes in graph, edge-aware node update (EANU) module updates the representation for each node by aggregating the dependency attention information from its neighbors through the attention output tensor. Mathematically, this operation can be defined as follows:

$$\begin{aligned} \mathbf{H}^l &= EANU(\mathbf{H}^{l-1}, \mathbf{E}^{l-1}) \\ &= \sigma(Pool(\mathbf{H}_1^l, \mathbf{H}_2^l, \ldots, \mathbf{H_p}^l)). \end{aligned}$$

Specifically, the dependency attention aggregation is conducted as follows:

$$\mathbf{H}^l = (\mathbf{H}^{l-1} + \mathbf{O}_{sd}^{l-1} + \mathbf{O}_{od}^{l-1})\mathbf{W}.$$

We followed the node-aware edge update (NAEU) module proposed in EE-GCN (Cui et al., 2020b) to dynamically calculate and update edge representations according to the node context. Formally, the NAEU operation is defined as:

$$\begin{aligned} \mathbf{H}^l &= NAEU(\mathbf{E}_{i,j,:}^{l-1}, \mathbf{h}_i^l, \mathbf{h}_j^l) \\ &= \mathbf{W}_u[\mathbf{E}_{i,j,:}^{l-1} \oplus \mathbf{h}_i^l \oplus \mathbf{h}_j^l], i, j \in [1, n]. \end{aligned}$$

For the original EE-GCN, it will contain noisy edges ,but adding attention features will pay much attention to the edges which denoting the latent dependency relations. However, the drawback of such an update method is that as the depth of the network increases, updated by the attention tensor, the initialized node-to-node feature plays a smaller role during training process. The specific details will be introduced in the experiment section below

4 Experiments

4.1 Dataset, preprocessing and Evaluation Metrics

We conduct experiments on the ACE2005 dataset, which is the standard supervised dataset for event detection. The Stanford CoreNLP toolkit3 [†] is used for dependency parsing. ACE2005 contains 599 documents annotated with 33 event types. We use the same data split as previous works ((Chen et al., 2015); (Nguyen et al., 2016); (Liu et al., 2017); (Yan et al., 2019); (Cui et al., 2020b) for train, dev and test set, and describe the details in the supplementary material (Data.zip). We performed the event detection task on the sentence-level data, calling us to preprocess the datatset on the sentence level. Thanks to the annotations of the ACE2005 data set, each sentence has a corresponding sentence-id, and each event also has an event-id. We finally merged different event triggers and dependency labels of a same sentence

[†] http://nlp.stanford.edu/software/stanford-english-corenlp-2018-10-05-models.jar

together. Resembling previous work, we evaluated our model via the official scorer in terms of the Precision (P), Recall (R) and F1-score.[‡]

4.2 Comparing with State-Of-The-Art Method

We report our experimental results on the ACE2005 dataset in Table 1. It is shown that our model, SA-GCN, outperforms the baselines of the original EE-GCN and achieves state-of-the-art F1-score with the help of the self-attention mechanism. We attribute the performance gain to two aspects: 1) The introduction of self-attention mechanism. Unlike most of the exising GCN based method, we introduced self-attention mechanism to perform event detection. Through three learnable parameters to update a attention tensor based on the initialized asymmetric adjacency tensors. Both of the syntactic relationship-based graph structure and the typed dependency labels are applied to our network structure. The attention matrix is updated through three learnable parameters, and the node-to-edge features supplemented by different degrees of attention are fused with the node-to-node features encoded by word embedding and entity type embedding. As a result, SA-GCN baseline outperforms EE-GCN 2.1% in Precision score and simultaneously keep the same performance on the Recall score. Such a performance improvement validates that self-attention mechanism can indeed explore more potential dependency relations based on the EE-GCN and has a good performance in event trigger detection. 2) The introduction of the asymmetric tensor for edge representation. Compared with the symmetric adjacency tensor in EE-GCN, we use two asymmetric adjacency tensors SUB and OBJ, to encode the input of the Self-Attention module. This design makes the performance of SA-GCN more prominent in the experiment of residual network.

Due to the limitation of GPU memory, we tested the 4-layer GCN baseline and the network structure using GResNet. The baseline results show that as the depth of the network increases, the performance of the two networks will decrease instead. In particular, since the adjacency tensor of EE-GCN represents the structure of the entire graph, the Recall score will continue to rise. However, the application of such a dependency-based topology approach will reduce the prediction of specific event triggers. Due to the introduction of the self-attention mechanism, our SAEE-GCN pays more attention to the latency dependency relations between word sequences, which helps to improve the prediction of event trigger words,but meanwhile, the initialized node features has less and less influence on attention tensor after continuous iteration and update.

Based on this comparison, we try to add residual terms to EE-GCN and SA-GCN to further realize the expression of word dependence in sentences. In the experiment, we set the hyperparameters α to weigh the performance of the network structure in Precision Score and Recall Score. When $\alpha = 0.8$, the performance reaches the best.

Baseline (Cui et al., 2020b)

2-layer	**P**	**R**	F1
EE-GCN	75.94	77.86	76.89
SA-GCN	78.03	77.56	77.79
4-layer			
EE-GCN	69.12	**82.47**	75.2
SA-GCN	**81.80**	68.80	74.74
4-layer GResNet			
α**=0.8**			
EE-GRCN	76.22	71.92	74.01
SA-GRCN	78.58	77.41	**77.99**
α**=0.5**			
EE-GRCN	76.40	75.04	75.71
SA-GRCN	73.97	80.24	76.98

Table 1: SA-GRCN results on ACE2005.

Model	**Dev** F1
Best SA-GRCN	68.09
-GRT	67.32
-SA	66.98
-EANU	67.49
-NAEU	67.9

Table 2: An ablation study on SA-GRCN. GRT is short for the Graph Residual Term. SA is short for Self-Attention collect module. EANU and NAEU is short for edge-aware node update module and node-aware edge update respectively.

4.3 Ablation Study

In order to prove the effectiveness of each component, we conducted an ablation study on the ACE2005 dev set. As shown in Table 2: 1) Graph residual term (GRT): In order to study whether

[‡] https://github.com/yubochen/NBTNGMA4ED/

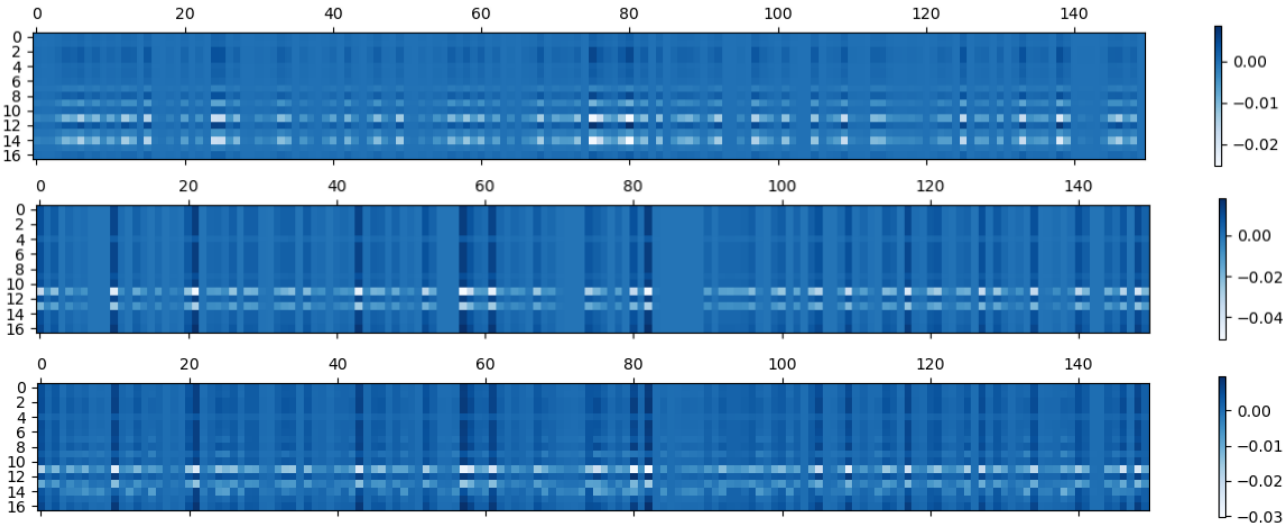

Figure 3: A visualization of the outputs of Self-Attention collect module encoded by **SUB** and **OBJ** . The example sentence is the same as the sentence shown in Fig1

GResNet can improve the overall performance, we set GresNet's hyperparameter $\alpha = 1$ to eliminate graph residual term. As a result, the F1 score dropped by 0.77%, which shows that GResNet plays an important role in SA-GRCN. According to the comparison with the SA-GCN, SA-GRCN largely balances the loss of node features caused by the self-attention module. 2) Self-attention collect module (SA): removing the self-attention collect module means our SA-GRCN degraded to an architecture similar to the EE-GCN, and the performance will reduce by 1.1% consequently, which validify the effectiveness of the self-attention mechanism for exploring latent dependency relations. 3) Edge-Aware Node Update module(EANU) and Node-Aware Edge Update Module (NAEU): comparing with the other two module, update methods influence less. However, we noticed that the impact of EANU on the performance of SA-GRCN is 0.3% greater than that of NAEU. As analyzed in the previous section, comparing with the stationary adjacency tensor, our proposed attention tensor allow the EANU to mine latent dependency relations, which again confirms the effectiveness of our model.

5 Effectiveness of Self-Attention collect module

As shown in Fig3, the overall outputs of the self-attention collect module can be divided into the **SUB** attention output and **OBJ** attention output. Since the hiddern layer of the Self-Attention collect module represent the latent dependency relations, the dimension of the output is $n \times dim$. Experiment results show that the SUB attention output of the 9th node "hospital" and 13th node "deaths" are marked as dark blue, which confirms that our self-attention module explore the key node "hospital" to detect event "Die". We also notice that "Medical Center" which directly connected to the root node in Fig1, weigh less than the key node "hospital". In **OBJ** attention output, event trigger "wounded" is given a high attention weight to explore latent dependency relations. What's more, noisy node like "," and "with" as we mentioned above, are marked in light blue or white, representing a lower attention weight. Finally, both "Die" and "Injure" are detected by SA-GRCN.

6 Conclusions and Future Works

In this paper, we propose a novel model named Self-Attention Graph Residual Convolutional Networks (SA-GRCN) for event detection. SA-GRCN introduces the typed dependency label information into the graph modeling process, and learns to update the relation representations in a context-dependent manner. Experiments show that our model achieves the start-of-the-art results on the ACE2005 dataset. However, the shortcomings of our architecture is that as the depth of the network layer increases, the introduction of the attention mechanism can greatly improve the accuracy of prediction and reduce the recall score. Currently, GResNet provides a solution that via introducing manually defined residual

term and hyperparameter α to balance the overall performance of the model. Our follow-up research work is through mathematical modelling analysis to find a more suitable residual term, and set α to be a learnable parameter.

References

Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. Automatically labeled data generation for large scale event extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–419, Vancouver, Canada. Association for Computational Linguistics.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling 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 167–176, Beijing, China. Association for Computational Linguistics.

Yubo Chen, Hang Yang, Kang Liu, Jun Zhao, and Yantao Jia. 2018. Collective event detection via a hierarchical and bias tagging networks with gated multi-level attention mechanisms. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1267–1276, Brussels, Belgium. Association for Computational Linguistics.

Shiyao Cui, Bowen Yu, Xin Cong, Tingwen Liu, Quangang Li, and Jinqiao Shi. 2020a. Label enhanced event detection with heterogeneous graph attention networks. *CoRR*, abs/2012.01878.

Shiyao Cui, Bowen Yu, Tingwen Liu, Zhenyu Zhang, Xuebin Wang, and Jinqiao Shi. 2020b. Edge-Enhanced Graph Convolution Networks for Event Detection with Syntactic Relation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2329–2339, Stroudsburg, PA, USA. Association for Computational Linguistics.

Shaoyang Duan, Ruifang He, and Wenli Zhao. 2017. Exploiting document level information to improve event detection via recurrent neural networks. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 352–361, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Xiaocheng Feng, Lifu Huang, Duyu Tang, Heng Ji, Bing Qin, and Ting Liu. 2016. A language-independent neural network for event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 66–71, Berlin, Germany. Association for Computational Linguistics.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition.

Yu Hong, Wenxuan Zhou, Jingli Zhang, Guodong Zhou, and Qiaoming Zhu. 2018. Self-regulation: Employing a generative adversarial network to improve event detection. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 515–526, Melbourne, Australia. Association for Computational Linguistics.

Viet Dac Lai, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020. Event detection: Gate diversity and syntactic importance scores for graph convolution neural networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5405–5411, Online. Association for Computational Linguistics.

Shulin Liu, Yubo Chen, Shizhu He, Kang Liu, and Jun Zhao. 2016. Leveraging FrameNet to improve automatic event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2134–2143, Berlin, Germany. Association for Computational Linguistics.

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1789–1798, Vancouver, Canada. Association for Computational Linguistics.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256, Brussels, Belgium. Association for Computational Linguistics.

Yaojie Lu, Hongyu Lin, Xianpei Han, and Le Sun. 2019. Distilling discrimination and generalization knowledge for event detection via delta-representation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4366–4376, Florence, Italy. Association for Computational Linguistics.

Jonas Mellin and Mikael Berndtsson. 2009. *Event Detection*, pages 1035–1040. Springer US, Boston, MA.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association*

for Computational Linguistics: Human Language Technologies, pages 300–309, San Diego, California. Association for Computational Linguistics.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web*, pages 593–607, Cham. Springer International Publishing.

Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. 2020. Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6578–6588, Online. Association for Computational Linguistics.

Wei Xiang and Bang Wang. 2019. A survey of event extraction from text. *IEEE Access*, 7:173111–173137.

Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. 2019. Event detection with multi-order graph convolution and aggregated attention. 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 5766–5770, Hong Kong, China. Association for Computational Linguistics.

Fan Yang, Xiaochang Peng, Gargi Ghosh, Reshef Shilon, Hao Ma, Eider Moore, and Goran Predovic. 2019. Exploring deep multimodal fusion of text and photo for hate speech classification. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 11–18, Florence, Italy. Association for Computational Linguistics.

Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. 2018. Graph r-cnn for scene graph generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Jiawei Zhang and Lin Meng. 2019. Gresnet: Graph Residual Network for Reviving Deep Gnns From Suspended Animation. *arXiv*, (2016):1–18.

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

Yue Zhao, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. 2018. Document embedding enhanced event detection with hierarchical and supervised attention. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 414–419, Melbourne, Australia. Association for Computational Linguistics.