Received August 28, 2020, accepted September 12, 2020, date of publication September 18, 2020, date of current version September 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3024872

Graph Convolution Over Multiple Latent Context-Aware Graph Structures for Event Detection

LEI LI[1,2,3], LI JIN[1,2], ZEQUN ZHANG[1,2], QING LIU[1,2], XIAN SUN[1,2,3], AND HONGQI WANG[1,2,3]

[1]Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100190, China
[2]Key Laboratory of Network Information System Technology, Institute of Electronics, Chinese Academy of Sciences, Beijing 100190, China
[3]School of Electronic, Electrical, and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100190, China

Corresponding author: Li Jin (jinlimails@gmail.com)

ABSTRACT Event detection is a particularly challenging problem in information extraction. The current neural network models have proved that dependency tree can better capture the correlation between candidate trigger words and related context in the sentence. However, syntactic information conveyed by the original dependency tree is insufficient for detecting trigger since the dependency tree obtained from natural language processing toolkits ignores semantic context information. Existing approaches employ a static graph structure based on original dependency tree which is incompetent in terms of distinguishing interrelations among trigger words and contextual words. So how to effectively make use of relevant information while ignoring irrelevant information from the dependency trees remains a challenging research question. To address this problem, we investigate a graph convolutional network over multiple latent context-aware graph structures to perform event detection. We exploit a multi-head attention mechanism on BERT representation and original adjacency matrix to generate multiple latent context-aware graph structures (a "dynamic cutting" strategy), which can automatically learn how to select the useful dependency information. Furthermore, we investigate graph convolutional networks with residual connections to combine the local and non-local contextual information. Experimental results on ACE2005 dataset show that our model achieves competitive performances compared with the methods based on dependency tree for event detection.

INDEX TERMS Event detection, graph convolutional network, multi-head attention.

I. INTRODUCTION

Event detection is an important and challenging task in information extraction, which aims to discover specific types of event triggers in texts. This task is one of the most significant research topics in social media analysis [1], which has been beneficial to a wide range of downstream tasks, including question answering [2], automatic text summarization [3] and others. Event triggers are generally verbs or nominalizations that evoke the events of interest, which serve as the main word(s) to the corresponding event. The event detection task, more precisely stated, involves how to identify event triggers and classify them into specific types. An example is shown in Figure 1, in the sentence *"In Baghdad, a cameraman died when an American tank fired on the Palestine Hotel"*, an event detection system is expected to detect *"died"* as a trigger of type **Die** and *"fired"* as a trigger of type **Attack**.

In early years, researchers had designed various hand-crafted feature sets such as lexical features and contextual features to extract events [4]–[6]. Although these methods can achieve high performance in specific fields, they rely heavily on manual annotations and features specific for each event type. Recently, most existing event extraction models can be categorized into two classes: sequence-based models and dependency-based models. Sequence-based models only use the given sentences [7]–[9] whereas dependency-based models incorporate dependency trees into the models [10]–[14]. Compared to sequence-based models, dependency-based models can more effectively obtain the connection between the event trigger word and its corresponding entity or other trigger words through the non-local syntactic structure in the sentence.

The associate editor coordinating the review of this manuscript and approving it for publication was Orazio Gambino.

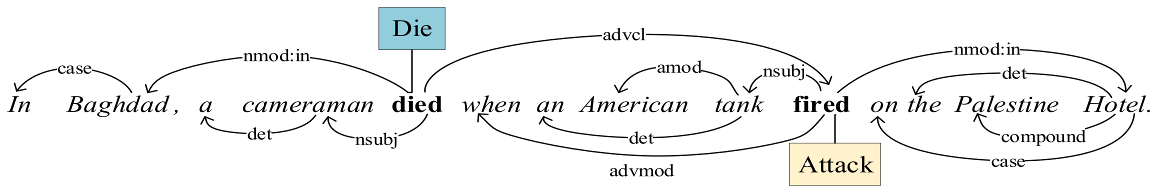
Die
Attack
In Baghdad , a cameraman died when an American tank fired on the Palestine Hotel.
case
nmod:in
det
nsubj
advcl
amod
nsubj
det
advmod
nmod:in
det
compound
case

FIGURE 1. An example of syntactic dependency parsing result produced by Stanford CoreNLP. The sentence contains two events: a Die event triggered by the word *died* and an Attack event triggered by the word *fired*.

Among dependency-based models, Graph Convolution Networks (GCNs) [15] is often performed to model syntactic dependency information for event detection. For example, reference [11] constructs a graph structure using the adjacency matrix obtained from the dependency tree. Multi-order syntactic representations in sentences are utilized via graph convolution network with aggregative attention [13]. However, the above dependency-based models explicitly use a invariant graph structure generated by the adjacency matrix and they can't distinguish which syntactic information is useful or useless from the dependency tree. Figure 1 depicts the dependency parse tree of the aforementioned sentence. Words along the *"fired"* form a trimmed phrase "*American tank fired on Hotel*" of the original sentence, which conveys much information about the candidate trigger *"fired"*. In other words, such as the rest "*In Baghdad, a cameraman died when*" is less informative and may bring noise if not dealt with properly. Intuitively, a model, capable of learning to maintain a balance between including and excluding information in the full tree, will benefit event trigger detection.

To alleviate the problems above, we propose a graph convolutional network over multiple latent context-aware graph structures for event detection. Firstly, a sentence is fed into BERT [16] to generate BERT representation. Secondly, based on the BERT representation, multi-head attention [17] is employed to substitute the original dependency graph into multiple fully connected edge-weighted graphs. Then, GCNs take BERT representations and adjacency matrixs based on fully connected graphs as input to learn syntactic contextual representation of each node. Residual connection [18] is employed to integrate BERT representation with syntactic contextual representations. Finally, a linear classifier is adopted to extract event triggers. The multi-head attention mechanism exploits BERT representations to transform the original syntactic graph into multiple latent context-aware graphs, which can be understood as a "dynamic cutting" strategy on dependency matrix. Multi-head attention jointly attending to information from different representation subspaces at different positions allows the model to learn how to select relevant sub-structures for candidate triggers. Through such "dynamic cutting" strategy, GCNs can draw efficiently syntactic structure information. What's more, the residual connection mechanism combines BERT representation and GCNs information to better leverage the structural information of the full dependency tree in event detection task.

We extensively evaluated the effectiveness of our model on the ACE2005 dataset. Experimental results show that our model outperforms previous dependency-based approaches in terms of both Recall and F1-measure. In summary, the contributions of this article are as follows:

- We propose a new method for event detection with a "dynamic cutting" strategy. It utilizes BERT semantic representation to generate multiple latent context-aware graph structures via multi-head attention mechanism. Such a "dynamic cutting" strategy on adjacency matrix learns how to select and discard information in the dependency trees at different positions.
- We employ Graph Convolutional Network to exploit more non-local and non-sequential dependency information. Furthermore, a residual connection mechanism is investigated to incorporate the local contextual representations learned by BERT with the non-local contextual information generated by Graph Convolutional Networks.
- Experiment results show that our proposed method can substantially improve the performance of event detection, where the F1-measure and Recall score achieve 76.34%, 75.21% on ACE2005 dataset, respectively.

II. RELATED WORK

Graph Convolutional Network has been employed in event detection to explore the syntactic representation, which provides an effective mechanism for linking words directly to their informative contexts in the sentence. In this work, we divide the relevant studies into following two categories, namely event detection and graph convolutional networks.

A. EVENT DETECTION

Event detection is a fundamental problem in information extraction and natural language processing [7], [19]. The early approach for event detection involved the feature-based methods which employed hand-design feature sets in different statistical models [5], [20]. The feature-based methods required extensive human engineering which essentially affects model performance. The last couple of years witnessed the success of neural network models for event detection. The typical models employed convolutional neural networks (CNNs) [7], [21], recurrent neural networks (RNNs) [8], [22]. While such models effectively captured relations in the local context, they had limited capability of exploiting non-local and non-sequential dependencies. In many applications, however, such dependencies could significantly reduce tagging ambiguity and improve overall performance.

In order to capture non-local dependencies in the input space, dependency-based approaches were exploited to incorporate structural information into neural models. Syntactic relation representation based on dependency trees could better capture the interrelation between candidate trigger words and related contexts than sentence representation. A novel dependency bridge recurrent neural network (dbRNN) was proposed to carry syntactically related information when modeling each word for event extraction [10]. Reference [11] used CNN based on dependency tree and an entity mention-guided pooling scheme to perform event detection. Syntactic shortcut arc was introduced to enhance information flow and attention-based GCN to perform event detection [12]. Reference [13] proposed a dependency tree based GCN model with aggregative attention to combine multi-order word representations from different GCN layers. However, these GCN-based models are powerless in distinguishing messy syntactic information due to the invariant dependency graph structure. In this article, we propose a model to select useful information dynamically from dependency tree by which the performance can be improved.

B. GRAPH CONVOLUTIONAL NETWORKS

Graph convolutional networks generalize CNNs over graphs [23], [24], which is one of the typical variants of graph neural networks (GNN). It has been successfully applied to many natural language processing tasks, such as text classification [25], semantic role labeling [26], relation extraction [27], [28], machine translation [29] and knowledge base completion [30]. Early efforts [23], [31] attempted to extend neural networks to deal with arbitrary structured graphs, where the states of nodes are updated based on the states of their neighbors. Given a graph, a graph convolutional network can embed the node by recursively aggregating the node representation of its neighbors. Subsequent efforts improved its computational efficiency with local spectral convolution techniques [32]. Our method is closely related to the GCNs [11], which used a convolutional neural network based on dependency trees and a novel pooling method to improve the performance of event detection task.

More recently, graph attention networks (GATs) [33] was proposed to summarize neighborhood states by using masked self-attentional layers [17]. Compared with their work, the motivation and method when applying an attention mechanism in graph convolutional networks are different. Particularly, each node only focuses on its neighbors in GATs, while our model involves the correlation among all nodes. The network topology in GATs remained unchanged, while our model will construct fully connected graphs to capture long-range semantic interactions.

TABLE 1. Automatic content extraction terminologies.

Terminology	Description
Entity	an object or a set of objects in one of the semantic categories of interests
Entity mention	a reference to an entity (typically, a noun phrase)
Event trigger	the main word that most clearly expresses an event occurrence
Event argument	the mention that is involved in an event (participant)
Event mention	a phrase or sentence within which an event is described including the trigger and arguments

TABLE 2. Example of tag assigning scheme in the sentence.

words	···	*a*	*cameraman*	*died*	*when*	···
label	···	**O**	**O**	**Die**	**O**	···

III. TASK DEFINITION

In this article, we focus on event detection task defined in Automatic Content Extraction (ACE) evaluation, where an event is defined as a specific occurrence involving one or more participants. We firstly introduce some ACE terminologies to facilitate the understanding of this task in Table 1.

The goal of event detection is to identify event triggers and categorize their event types. For example, in the sentence *"Obama beats McCain."*, an event detection system is expected to detect an **Elect** event along with the trigger word *"beats"*. The ACE 2005 evaluation defines 8 super types of events, with 33 subtypes and a "Not Applicable(*NA*)" type, such as **Attack** or **Elect**. Following previous works [7], [8], [19], [34], we treat these simply as 34 separate event types and ignore the hierarchical structure among them.

There are triggers that consist of multiple tokens. The previous work [35] applies a BIO annotation schema to assign trigger label to each token w_i. In this work, we treat consecutive tokens which share the same predicted label as a whole trigger.

IV. METHODOLOGY

Event detection can be formalized as a sequence labeling problem. We treat every word in a sentence as a trigger candidate, and classify each candidate to a certain event type. As shown in Table 2, the event detection model collectively assigns a tag for each word in the sentence to indicate whether it triggers a specific type of event (**O** means it does not belong to any event type).

In this section, we illustrate the details of the proposed method for event detection. Figure 2 shows the overall architecture of our proposed model, which primarily involves the following four components:

(I) **Sentence Encoding Layer** which encodes input sentence to a sequence of hidden embeddings.

(II) **Graph Construction Layer** which exploits a multi-head attention mechanism to construct multiple latent context-aware graph structures.

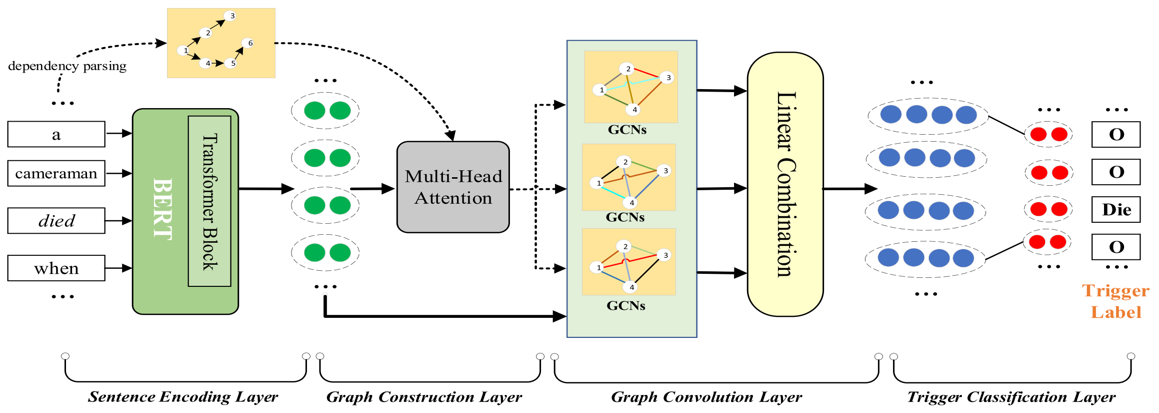

FIGURE 2. Model architecture shown with an example sentence *"In Baghdad, a cameraman died when an American tank fired on the Palestine Hotel"*, where some words are omitted. The dotted line represents information flow of the adjacency matrix and $N = 3$ (N is num of multi-head).

(III) **Graph Convolution Layer** which performs graph convolution over multiple connected graphs with the help of residual connections.

(IV) **Trigger Classification Layer** which assigns an event type (including the *NA* type) to each candidate in a sentence.

In the following section, we will introduce them respectively.

A. SENTENCE ENCODING LAYER

Formally, an event detection instance is an n-word sequence $W = \{w_1, \ldots, w_n\}$, where w_i refers to the i-th token. Sentence encoder is adopted to encode the word sequence W into hidden embeddings,

$$\{h_1, h_2, \ldots, h_n\} = E\{w_1, w_2, \ldots, w_n\} \tag{1}$$

where $E(\cdot)$ is a neural network to encode the sentence. In this article, we select BERT [16] as sentence encoder.

The input of the BERT is the concatenate of three types of embedding, including WordPiece embedding [36], position embedding and segment embedding. The input sequence for BERT is constructed as follows:

$$S = [< CLS >, sentence, < SEP >] \tag{2}$$

where $< CLS >$ and $< SEP >$ are special tokens of BERT. Since the input contains only one sentence, all its segment ids are set to zeros.

After feeding the input representation described above into 12 successive Transformer encoder blocks, we can obtain the BERT contextual representation $H = \{h_0, h_1, \ldots, h_n, h_{n+1}\}$. The final remaining $H = \{h_1, h_2, \ldots, h_n\}$ is the sequence of hidden embeddings while removing the representation of $< CLS >$ and $< SEP >$, which will be used as an input of the subsequent graph construction layer.

B. GRAPH CONSTRUCTION LAYER

Each dependency tree can be represented as a syntactic graph in terms of an adjacency matrix. Let $\boldsymbol{A}$ be the adjacency matrix of original syntactic graph, which is generated from dependency tree of the sentence. As mentioned before, pre-existed methods maintain the invariance of the original syntactic graph structure. However, in graph construction layer, BERT representation and adjacency matrix based on original dependency tree are used to generate multiple latent context-aware graph structures. The original dependency tree is substituted into a fully connected edge-weighted graph by constructing an attention guided adjacency matrix $\tilde{A}$. In this work, we compute $\tilde{A}$ by using multi-head attention [17]. The model employs BERT semantic vectors to calculate the correlation between words from different perspectives (N head). An attention function can be described as mapping a query and a set of key-value pairs to the output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key:

$$\tilde{A}^{(t)} = softmax(\frac{QW_t^Q \times (KW_t^K)^T}{\sqrt{d}})V \tag{3}$$

where t is the t-th head ($t = 1, \cdots, N$), Q and K are both equal to the output of sentence encoding module H, $W_t^Q \in \mathbb{R}^{d*(d/N)}$ and $W_t^K \in \mathbb{R}^{d*(d/N)}$ are the parameter matrices, d is the dimension of hidden layer. After using multi-head attention mechanism, the system can get N different attention guided adjacency matrices, where N is the head. $\tilde{A}^{(t)}$ is the t-th attention oriented adjacency matrix generated by the t-th head.

Figure 3 shows an example of transforming the original adjacency matrix into multiple attention guided adjacency matrices. BERT representation and the adjacency matrix are fed into multi-head attention mechanism to produce multiple fully connected edge-weighted graphs. Compared with the original sparse dependency graph, the fully connected edge-weighted graphs can better express the correlation between nodes. In practice, we treat the original adjacency matrix as an initialization so that the dependency information can be captured in the node representations for later

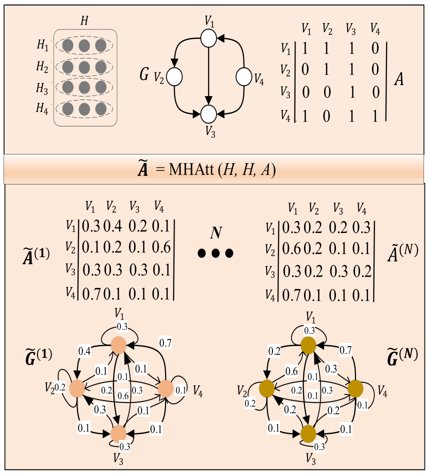

FIGURE 3. **Graph Construction Layer. *N* attention guided adjacency matrices ($\tilde{A}$) are constructed by using multi-head attention. *H* indicates BERT representation and MHAtt is the multi-head attention mechanism. These weights in matrices are viewed as the strength of relatedness between nodes.**

attention calculation. The attention guided adjacency matrices and BERT embeddings are jointly into GCNs for learning syntactic contextual representation of each word.

C. GRAPH CONVOLUTION LAYER

This part uses graph convolutional network with residual connection over multiple latent context-aware graph structures to model text sentences.

1) GCNs

We firstly introduce the general framework of graph convolutional network. Given a graph with n nodes, an $(n \times n)$ adjacency matrix $\boldsymbol{A}$ can represent the graph. Let $\mathcal{G} = \{V, \mathcal{E}\}$ be the dependency parse tree for a sentence $W = \{w_1, \ldots, w_n\}$, where V contains n nodes corresponding to the n tokens in W and $\mathcal{E}$ are the sets of edges. Each edge (w_i, w_j) in $\mathcal{E}$ is directed from the head word w_i to the dependent word w_j and has the dependency label $K(w_i, w_j)$. Following the previous work [15], we also add all the self-loops (w_i, w_i).

For instance, in the dependency tree of Figure 1, there is a directed edge from the node for word w_i = *"died"* (the head word) to the node for word w_j = *"cameraman"* (the dependent word) with the edge label $K(w_i, w_j) = K($*"died", "cameraman")* = *nsubj*, the reversed dependency arc with the additional type label $K($*"cameraman", "died")* = *nsubj'*, and two self-loops of *"died"* and *"cameraman"* with type label $K($*"died", "died")* $= K($*"cameraman", "cameraman")* = *loop*.

Each graph structure $\mathcal{G}$ based on dependency tree corresponds to an adjacency matrix $\boldsymbol{A}$. Intuitively, $a_{ij} = 1$ and $a_{ji} = 1$ if an edge exists between node i and node j, otherwise $a_{ij} = 0$ and $a_{ji} = 0$, where a_{ij} is an element in $\boldsymbol{A}$.

Graph convolution operation aims to gather information from neighbor nodes in the graph. In particular, the graph convolution vector $h_i^{(l)}$ at the l-th layer for node i can be computed as follows:

$$h_i^{(l)} = \rho(\sum_{j=1}^{n} A_{ij} W^{(l)} h_j^{(l-1)} + b^{(l)}) \tag{4}$$

where A is the adjacency matrix of a graph with n nodes, $W^{(l)}$ and $b^{(l)}$ are the model parameters, and ρ is an activation function (e.g. RELU [37]). Moreover, we use the output of the sentence encoding module H to initialize node representation $h^{(0)}$ of the first layer of GCNs. N separate graph convolutional operations are required as we have N different fully-connected adjacency matrices.

2) GCNs WITH RESIDUAL CONNECTIONS

GCNs can get information about the k hops by stacking k-layers, but sometimes the length of shortcut path between two triggers is less than k. To avoid information over-propagating, we adapt residual connections [18] into GCNs. A residual connection mechanism can incorporate the local contextual representation learned by BERT with the non-local contextual information generated by Graph Convolutional Networks. Note that the application of this mechanism can allow contextual information flowing across GCN layers. Specifically, the initial representation of GCN can be integrated into node's representation on the last iteration. Based on Equation (4), the computation of L-th layer (L is the total layers of GCNs) is modified as follows:

$$h_i^{(L)} = \rho(\sum_{j=1}^{n} A_{ij} W^{(L)} h_j^{(L-1)} + b^{(L)}) + h_i^{(0)} \tag{5}$$

where $h^{(0)}$ is the initial input vector (the BERT representation H) and the other parameters are defined as above. The BERT representation is only integrated into the last iteration while graph representation updates depending on Equcation (4) in other layers. Based on the t-th attention guided adjacency matrix $\tilde{A}^{(t)}$ $(t = 1, \cdots, N)$, we can get the final output for the t-th head (H_{t_i} is the final representation of i-th token):

$$H_t = [H_{t_1}, H_{t_2}, \cdots, H_{t_n}] \tag{6}$$

3) GCNs WITH LINEAR COMBINATION

In addition to residually connected layers, we include a linear combination layer after multi-layer GCNs to merge the representations from different GCN blocks, reaching a more expressive representation. In which the final representation of each node is computed by fusing the node's representations from all graph convolution networks. Formally, the outputs of multiple GCNs are concatenated and feed into the linear combination layer as follows:

$$D_{com} = W_{com} H_{con} + b_{com} \tag{7}$$

where H_{con} is the concatenation of N separate GCNs outputs, $H_{con} = [H_1, \cdots, H_N] \in \mathbb{R}^{d*N}$, N is num of multi-head. $W_{com} \in \mathbb{R}^{(d*N)*d}$ is a learnable transformation matrix and b_{com} is the bias vector.

TABLE 3. Statistics of the ACE2005 dataset.

	Train	Dev	Test
Documents	529	40	30
Sentences	14,837	863	672
Triggers	4,337	492	422

D. TRIGGER CLASSIFICATION LAYER

After applying the proposed method over the GCNs, the syntactic contextual representations of all tokens are obtained. The goal of event detection is to predict the type of tokens based on these representations. The context vector D_{com} are fed into a fully connected network to predict the trigger label.

$$v_i = f(w_d d_i + b_d) \tag{8}$$

$$y_i = softmax(W_v v_i + b_v) \tag{9}$$

where f is a non-linear activation, $d_i \in D_{com}$ and y_i is the final output of the i-th trigger label.

We minimize the negative log-likelihood loss function to train our model. Due to the data sparsity in the ACE 2005 dataset, we adapt a bias loss function [35] to balance event label weights during training.

$$J(\theta) = -\sum_{i=1}^{D}\sum_{j=1}^{D_{i,w}} I(y_{t_i}) \cdot \log(p(y_{t_i}|\theta)) + \beta(1 - I(y_{t_i})) \cdot \log(p(y_{t_i}|\theta)) \tag{10}$$

where D is the number of sentences in training corpus. $D_{i,w}$ is the number of tokens in D_i, $I(y_{t_i})$ is an indicating function, if y_{t_i} is O, it outputs 1, otherwise 0; β is a hyper-parameter larger than 1.

V. EXPERIMENTS

A. DATASET, EVALUATION METRIC AND RESOURCES

We evaluate method on the ACE2005 dataset. To comply with previous work, we use a pre-defined split of the documents as the previous work [8], [10], [38], Table 3 shows the data statistics.

Similar to previous work [39]–[42], we use the same criteria to judge the correctness of each predicted event mention.

- Event Trigger Identification: A trigger is correctly identified if its offset matches a reference trigger.
- Event Type Classification: A trigger is correctly classified if both its offset and event type match a reference trigger.

We use Stanford CoreNLP toolkit to parse every sentence in the corpus, including tokenizing, sentence splitting and generating dependency parsing trees.

B. EXPERIMENT SETTING

We use $BERT_{base}$ in our experiments, which adopts multi-layer bidirectional transformers to encode the input sequence into hidden embeddings. For fine-tuning on the development sets, the batch size (BS) is selected from (16,32) and the learning rate (lr) is selected from (1e-5, 2e-5, 3e-5, 5e-5). We select the head (N) from (2,3,6), the GCN layer (L) from (2,3,4) simultaneously. Through preliminary experiments, we discover that the combination (BS = 16, lr = 2e-5, N = 3, L = 3) gives the best results on task of event detection.

In the graph convolution layer, we use a three-layer GCN with 768 hidden units, linear combination with 768*N hidden units, where N is the head and 300 hidden units for linear classification layer. Stochastic gradient descent over shuffled mini-batches with the Adadelta update rule [43] is used for training processes. We use ReLU [37] as our nonlinear activate function. We also set dropout rate to 0.2 and the bias loss parameter β to 5.

C. OVERALL PERFORMANCE

To demonstrate the effectiveness of our model (called **MH-GCN**), we take several previous classic works for comparison, and divide them into three categories: feature-based models, sequence-based neural network models, external resource-based models and GCN-based neural network models.

1) FEATURE-BASED MODELS

- **Cross-Event** [44], using document-level information to improve the performance of ACE event extraction.
- **MaxEnt** [19], only using lexical features, basic features and syntactic features designed by human.

2) SEQUENCE-BASED NEURAL NETWORK MODELS

- **DMCNN** [7], which exploits a dynamic multi-pooling convolutional neural network for event trigger detection.
- **dbRNN** [10], which adds dependency bridges with weight to BiLSTM for event extraction.
- **LearnToSelectED** [45], which features the automatic identification of important context words in the sentence.

3) EXTERNAL RESOURCE-BASED MODELS

- **ATT+FrameNet** [34], the attention-based model augmented with annotated data in FrameNet.
- **GMLATT** [46], which exploits the multi-lingual information for more accurate context modeling.
- **BERT+Boot** [38], which applys an adversarial training mechanism to enhance distantly supervised event detection models.
- **EKD** [47], which leverages external open-domain trigger knowledge to provide extra semantic support.

4) GCN-BASED MODELS

- **GCN-ED** [11], which investigates GCN on syntactic dependency tree and an argument pooling mechanism to improve performance.
- **JMEE** [12], which exploits GCN with a self-attention aggregation mechanism and highway network to improve performance of GCN for event extraction.

TABLE 4. Overall Performance on the blind test data. † using dependency arcs only, ‡ using dependency arcs and dependency relations simultaneously and "-" means not reported.

Types	Models	Trigger Identification(%)			Trigger Classification(%)		
		P	R	F_1	P	R	F_1
Feature-based	$MaxEnt$	76.9	65.0	70.4	73.7	62.3	67.5
	$Cross-Event$	-	-	-	68.7	68.9	68.8
Sequence-based	$DMCNN$	80.4	67.7	73.5	75.6	63.6	69.1
	$dbRNN^{‡}$	-	-	-	74.1	69.8	71.9
	$LearnToSelectED$	-	-	-	75.4	75.0	75.2
External Resource-based	$ATT+FrameNet$	-	-	-	76.8	67.5	71.9
	$GMLATT$	80.9	68.1	74.1	78.9	66.9	72.4
	$BERT+Boot$	-	-	-	77.9	72.5	75.1
	EKD	-	-	-	79.1	78.0	78.6
GCN-based	$RA-GCN^{‡}$	-	-	-	76.7	78.6	77.6
	$GCN-ED^{†}$	-	-	-	77.9	68.8	73.1
	$JMEE^{†}$	80.2	72.1	75.9	76.3	71.3	73.7
	$MOGANED^{†}$	-	-	-	79.5	72.3	75.7
	$MH-GCN^{†}$(Ours)	80.14	78.34	79.23	77.51	75.21	76.34

- **MOGANED** [13], which uses GCN with aggregated attention to combine multi-order word representation from different GCN layers.
- **RA-GCN** [48], which investigates a relation-aware GCN on the tree with syntactic dependency relation labels.

MH-GCN is a GCN-based model which is compared with the state-of-the-art methods in Table 4 on the blind test set. From the table, we can make the following observations.

Among all these methods, sequence-based neural network models beat feature-based models on F1-measure. Moreover, compared with sequence-based models, external resource-based models and GCN-based models achieve better performance (except LearnToSelectED). This phenomenon is not surprising. Sequence-based neural network models can automatically learn salient features in the data to achieve better performance. External resource-based models utilize more data from external resource, and GCN-based models incorporate structural information to achieve further improvement.

We conduct a t-test to compare the proposed method to the other methods on F1-measure. Compared with the best feature-based models, MH-GCN gains 7.5 F1-score improvement. What's more, our model is superior to all of the sequence-based baselines. The MH-GCN model achieves the best performance among the GCN-based models which employ dependency arcs only. MH-GCN can improve the best recall and F1 in the best reported method with dependency arcs only MOGANED by +2.9 and +0.5, respectively. This demonstrates the effectiveness of our method to exploit BERT representation to enhance graph convolution via the multi-head attention mechanism.

Compared with RA-GCN, our model tends to achieve higher precise score but lower F1-measure. We notice the RA-GCN model exploits dependency arcs and relations simultaneously based a relation-aware GCN while MH-GCN uses dependency arcs only. The dependency relation obtained from syntactic parsing toolkit is indeed noisy, which is responsible for the lower precision. Our model outperforms these external resource-based models except EKD, as it investigates a teacher-student model to distill open-domain trigger knowledge from WordNet. It demonstrates that external resources are useful to improve event extraction.

D. MODEL STABILITY ANALYSIS

To further explore the performance stability of our model, we performe a 5-fold cross-validation on the ACE2005 dataset in Table 5. The ACE2005 corpus includes 6 different domains: broadcast conversation (bc), broadcast news (bn), telephone conversation (cts), newswire (nw), usenet (un) and webblogs (wl). In order to maintain data consistency of different domain in each fold. We divide the data of each domain into five parts and splice together. Finally, we choose a different fold each time as the testing set and used the remaining four folds as the training set. In Table 5, the average of trigger identification is 75.16% (variance is 0.21) and the average of trigger classification is 70.9% (variance is 0.18). We can see that our model obtains a close performance in each fold when the data for each domain is consistent.

In general, multiple experiments can improve robustness of the model since a single experiment may not reflect the true performance. We choose different random seeds and conduct a 10-test experiments based on a pre-defined data split [38]. Figure 4 show the 10-test experiments on ACE 2005 dataset. The average of trigger identification is 79.23% (variance is 0.18) and the average of trigger classification is 76.34% (variance is 0.05), which demonstrates our model has

TABLE 5. 5-fold cross-validation results on the ACE-2005 dataset. TI: Trigger Identification, TC: Trigger Classification, Ave: Average.

Fold	TI(%)			TC(%)		
	P	R	F_1	P	R	F_1
Fold 1	81.7	68.6	74.6	77.8	65.3	71
Fold 2	74	76	75	69.5	71.3	70.4
Fold 3	74	76	75	69.6	71.4	70.5
Fold 4	76.4	74	75.2	72.2	69.9	71
Fold 5	81.1	71.4	76	76.5	67.4	71.6
Ave	77.44	73.2	75.16	73.12	69.06	70.9

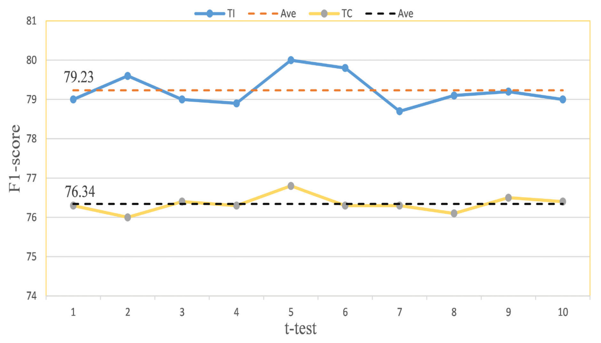

FIGURE 4. 10-test experiments on ACE 2005 dataset. TI: Trigger Identification, TC: Trigger Classification, Ave: Average.

strong robustness on event detection. Through two groups of experiments from different perspectives (i.e., cross-validation and t-test), it is obvious that the performance of our model is relatively stable.

E. EFFECT OF DEPENDENCY RELATION

To further reveal whether dependency relation can improve our modul, we design a model (MH-GCN-RW) based on MH-GCN. MH-GCN-RW: It applys gate on the edges to weight the importances of relation in GCN. Graph convolution operation is modified based on Equation (4) as follows:

$$h_i^{(l)} = \rho(\sum_{j=1}^{n} g_{ij}^{(l-1)}(A_{ij}W^{(l)}h_j^{(l-1)} + b^{(l)})) \tag{11}$$

$$g_{ij}^{(l)} = \sigma(r_{ij}W_{rel}^{(l)} + W_{rel}^{(l)}) \tag{12}$$

where $g_{ij}^{(l)}$ is the relation-weighted coefficient between nodes i and j at l-th layer, r_{ij} represents the relation embedding between nodes i and j, $W_{rel}^{(l)}$ and $b_{rel}^{(l)}$ are the model parameters according to relation, and other parameters are defined as above.

Table 6 shows the effectiveness of relation in our model. This modified model gets higher performance than MH-GCN while lower performance than RA-GCN. This demonstrates syntactic dependency relation can provide information to improve the performance. Note that MH-GCN-RW uses a simple fusion module while RA-GCN explores the relation-aware aggregation module and context-aware relation update module simultaneously. So designing an efficient relation fusion module will significantly improve the model. In the future, we are going to explore how to integrate syntactic dependency relation into our model.

TABLE 6. The effectiveness of dependency relation in MH-GCN.

Model	P	R	F1
RA-GCN	76.7	78.6	77.6
MH-GCN	77.5	75.2	76.3
MH-GCN-RW	77.9	76.2	77.0

TABLE 7. Performance of modified architectures based on MH-GCN.

Model	P	R	F1
BERT-GCN	75.3	74.2	74.7
MH-GCN/RC	74.1	72.5	73.2
MH-GCN-Mean	76.3	74.8	75.5
MH-GCN	**77.5**	**75.2**	**76.3**

F. ABLATION STUDY

To study the contribution of attention mechanism and residual connections, we design ablation experiments. For this purpose, we design three architectures based on MH-GCN: 1) BERT-GCN: it integrates BERT encoder with original syntactic graph structure instead of fully connected edge-weighted graphs; 2) MH-GCN/RC: it casts the residual connections away; 3) MH-GCN-Mean:it adopts mean pooling as the combination mechanism of multiple GCN blocks representations while MH-GCN adopts linear combination layer.

The experimental results are shown in Table 7. These three modified models all get lower performance than MH-GCN. MH-GCN/RC performs the worst, which suggests that residual connections mechanism plays an important role in context-aware graph structures. It implies that the context feature representation produced by BERT can bring essential information. Without residual connections, the GCNs converges slowly and most likely to ignore the relevant information. BERT-GCN drops more on precision than recall, which illustrates that multi-head attention learned from matrices helps MH-GCN to predict trigger words more precisely. The performance drop of MH-GCN-Mean is the smallest among the three modified models. Although the average of multiple GCNs representations achieves competitive performance for event detection, the proposed linear layer aggregation module still distinguishes the importance of syntactic representations of different head, which achieves 0.8% improvement on F1-measure.

G. EFFECT OF MULTI-HEAD GCNs ON EXTRACTING MULTIPLE EVENTS

In order to further prove the effectiveness of MH-GCN, especially for those sentences with multiple events, the test

TABLE 8. System performance on single event sentences(1/1) and multiple event sentences (1/N).

Stage	Model	1/1	1/N	all
trigger	Embedding+T	68.1	25.2	59.8
	CNN	72.5	43.1	66.3
	DMCNN	74.3	50.9	69.1
	JRNN	75.6	64.8	69.3
	JMEE	75.2	72.7	73.7
	MH-GCN	**78.9**	**74.2**	**76.3**

TABLE 9. Top two event types for three types of trigger errors.

Missing		Spurious		Wrong Type	
label	percent	label	percent	label	percent
Attack	8.11%	Attack	20.08%	Die	1.54%
Transport	5.79%	Transport	8.49%	Attack	0.77%
others	31.66%	others	20.08%	others	3.48%
total	45.56%	total	48.65%	total	5.79%

set is divided into two parts (single event and multiple events) following the previous work [8], [12] and perform evaluations separately. Single event means that one sentence only has one trigger; otherwise, multiple events in one sentences. Table 8 shows the performance (F1-measure scores) of DMCNN [7], JRNN [8], JMEE [12] and two other baseline systems, named Embeddings+T and CNN. Embeddings+T uses word embeddings and the traditional sentence-level features in [19] while CNN is similar to DMCNN, except that it applies the standard pooling mechanism instead of the dynamic multi-pooling method.

From the table, we see that MH-GCN significantly outperforms all the other methods when the input sentences contain more than one event (i.e. the row labeled with 1/N in the table). In the 1/N data split of triggers, our framework is 1.5% better than the JMEE, which demonstrates that our method uses multi-head attention to help alleviate multiple events issue. The multi-head attention mechanism allows the model to jointly attend to information from different representation subspaces at different positions, in order to maintain the correlation between multiple events.

H. ERROR ANALYSIS

We further examine output of MH-GCN on the test set to determine the contribution of each event type to trigger classification errors. Three types of errors have been arisen in this case: (i) missing an event trigger in the test set (called Missing), (ii) proposing a fakean event trigger (called Spurious), and (iii) misclassified event types (called Wrong Type). Table 9 shows the top two event types appearing in these errors and their corresponding percents. These three types account for 45.56% of the missing errors, 48.65% of the spurious errors and 5.79% of the wrong type errors. Attack and Transport are the types that are present frequently in misssing and spurious errors.

A careful analysis of the missing cases reveals that the errors mainly correspond to the trigger words not appearing in the training data, such as the word *"admits"* (of type **Transport**) in the sentence *"Turkey would lose a aid-package unless it admits troops into the country for the Iraq conflict."*. The spurious errors occur due to the confusable context of trigger words. For instance, the word *"war"* in the following sentence can be easily misproposed as an **Attack** event (due to its context with the word *"victims"*) *"...provided the money went for goods to victims of the first Gulf War."*. Highly ambiguous trigger (word triggers too many events) is the main reasons of the wrong type errors. In the sentence *"A rocket landed in farmlands and the other hit a house inside the refugee camp."*, the *"landed"* is misclassified as a **Transport** event instead of an **Attack** event.

For the problems mentioned above, our model may increase the scale of training data to solve rare or unseen trigger words by introducing external resources. Moreover, designing a model that can better capture modeling of the context will mitigate the confusable context problem.

VI. DISCUSSION

To better understand what the model has learned via the "dynamic cutting" strategy on adjacency matrix, we visualize the new edge-weighted adjacency matrix generated by attention mechanism based on BERT representation. We use a sentence *"In Baghdad, a cameraman died when an American tank fired on the Palestine Hotel"* as an example in Figure 5. These weights in adjacency matrix are viewed as the strength of relatedness between words.

Figure 5a shows the change of applying attention mechanism on dependency matrix. Since the original adjacency matrix generated by dependency tree is binary values (0 or 1) and symmetric matrices, edges on the top are the same color while are different color intensity on the bottom. Compared with the original dependency matrix, the pruned matrix of the attention mechanism can hierarchically express the correlation between words in the sentence. All words are equally important in the original dependency matrix although some words are directly connected by short arcs. However, the new adjacency matrices focus more on the words associated with the trigger words. For example, the edges in *died-cameraman*, *fired-tank* are darker than *in-cameraman*, *fired-Palestine* respectively. This shows attention mechanism has the ability to distill useful information for trigger word from the dependency tree.

Figure 5b demonstrates different dependency matrix generated by two attention heads. The first head gives more attention to the trigger word *fired* while the second exploits another trigger word *died* in the sentence. As the different attention head attends to information at different positions, the method of leveraging multi-head attention mechanism is helpful in alleviating the multiple events phenomenon. Multi-head attention can aggregate information from multi-dimensional

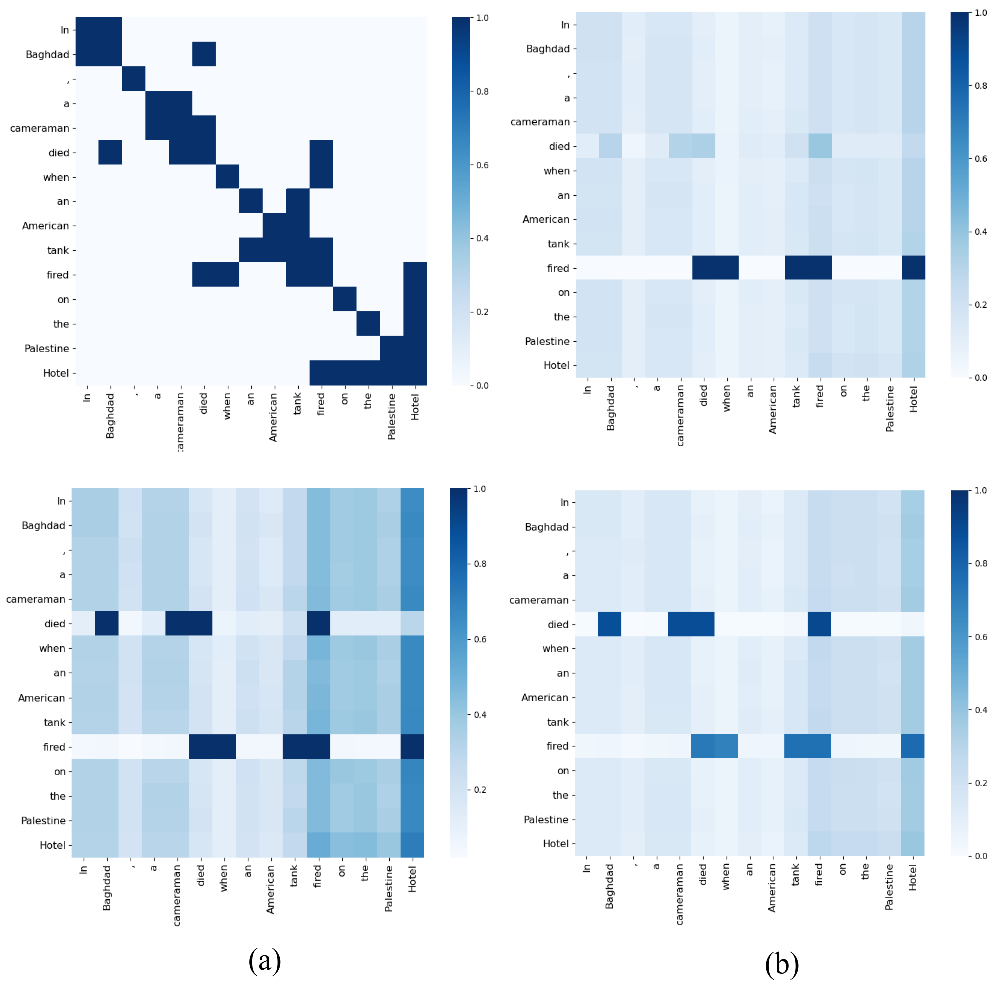

FIGURE 5. An example of the attention mechanism applying on dependency matrix in the graph construction layer. Dark colors represent closer syntactic connection between words. (a) The effects of attention mechanism. Top: the original dependency matrix. Bottom: new adjacency matrix generated by attention mechanism; (b) Two attention heads based on the original dependency matrix in multiple event. The top and bottom represent different heads.

space of multiple events to keep the associations between multiple events.

VII. CONCLUSION

We present a new method based on graph convolution network for event detection. An attention mechanism is applied on BERT representation and adjacency matrix to generate multiple latent context-aware graph structures, which can dynamically retain relevant information and ignore irrelevant information from the dependency trees at different positions. Using BERT semantic information for dynamically pruning on dependency matrices can distill more beneficial syntactic structure for trigger words. We introduce the graph convolutional networks with residual connections to combine the local and the non-local contextual information, which can effectively enhance the information flow of graph structure. The proposed model is empirically shown to be effective on the sentences with multiple events as well as yields the competitive performance on the ACE2005 dataset. For future work, we expect to investigate the joint models for event extraction (i.e. both event detection and argument prediction) based on the proposed model. We also plan to apply the GCNs models to the other datasets and extend it to other information extraction tasks such as relation extraction.

REFERENCES

[1] X. Chen, S. Wang, Y. Tang, and T. Hao, "A bibliometric analysis of event detection in social media," *Online Inf. Rev.*, vol. 43, no. 1, pp. 29–52, Feb. 2019.

[2] N. Kuchmann-Beauger, M.-A. Aufaure, and R. Thollot, "Context-aware question answering system," U.S. Patent 8 935 277, Jan. 13, 2015.

[3] L. Marujo, R. Ribeiro, A. Gershman, D. M. de Matos, J. P. Neto, and J. Carbonell, "Event-based summarization using a centrality-as-relevance model," *Knowl. Inf. Syst.*, vol. 50, no. 3, pp. 945–968, Mar. 2017.

[4] R. Grishman, D. Westbrook, and A. Meyers, "Nyu's english ace 2005 system description," in *Proc. ACE*, vol. 5, 2005.

[5] H. Ji and R. Grishman, "Refining event extraction through cross-document inference," in *Proc. ACL-HLT*, 2008, pp. 254–262.

[6] W. Lu and D. Roth, "Automatic event extraction with structured preference modeling," in *Proc. ACL*, 2012, pp. 835–844.

[7] Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao, "Event extraction via dynamic multi-pooling convolutional neural networks," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process.*, 2015, pp. 167–176.

[8] T. H. Nguyen, K. Cho, and R. Grishman, "Joint event extraction via recurrent neural networks," in *Proc. NAACL-HLT*, 2016, pp. 300–309.

[9] X. Feng, L. Huang, D. Tang, H. Ji, B. Qin, and T. Liu, "A language-independent neural network for event detection," in *Proc. ACL*, 2016, pp. 66–71.

[10] L. Sha, F. Qian, B. Chang, and Z. Sui, "Jointly extracting event triggers and arguments by dependency-bridge RNN and tensor-based argument interaction," in *Proc. AAAI*, 2018, pp. 5916–5923.

[11] T. H. Nguyen and R. Grishman, "Graph convolutional networks with argument-aware pooling for event detection," in *Proc. AAAI*, 2018, pp. 5900–5907.

[12] X. Liu, Z. Luo, and H. Huang, "Jointly multiple events extraction via attention-based graph information aggregation," 2018, *arXiv:1809.09078*. [Online]. Available: http://arxiv.org/abs/1809.09078

[13] H. Yan, X. Jin, X. Meng, J. Guo, and X. Cheng, "Event detection with multi-order graph convolution and aggregated attention," in *Proc. EMNLP-IJCNLP*, 2019, pp. 5770–5774.

[14] A. Balali, M. Asadpour, R. Campos, and A. Jatowt, "Joint event extraction along shortest dependency paths using graph convolutional networks," 2020, *arXiv:2003.08615*. [Online]. Available: http://arxiv.org/abs/2003.08615

[15] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: http://arxiv.org/abs/1609.02907

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: http://arxiv.org/abs/1810.04805

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[19] Q. Li, H. Ji, and L. Huang, "Joint event extraction via structured prediction with global features," in *Proc. ACL*, 2013, pp. 73–82.

[20] X. Li, T. H. Nguyen, K. Cao, and R. Grishman, "Improving event detection with abstract meaning representation," in *Proc. ACL-IJCNLP*, 2015, pp. 11–15.

[21] T. H. Nguyen and R. Grishman, "Event detection and domain adaptation with convolutional neural networks," in *Proc. ACL-IJCNLP*, 2015, pp. 365–371.

[22] A. N. Jagannatha and H. Yu, "Bidirectional RNN for medical event detection in electronic health records," in *Proc. NAACL-HLT*, 2016, p. 473.

[23] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Aug. 2005, pp. 729–734.

[24] F. Scarselli, M. Gori, A. Chung Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

[25] L. Yao, C. Mao, and Y. Luo, "Graph convolutional networks for text classification," in *Proc. AAAI*, 2019, pp. 7370–7377.

[26] D. Marcheggiani and I. Titov, "Encoding sentences with graph convolutional networks for semantic role labeling," in *Proc. EMNLP*, 2017, pp. 1506–1515.

[27] Y. Zhang, P. Qi, and C. D. Manning, "Graph convolution over pruned dependency trees improves relation extraction," in *Proc. EMNLP*, 2018, pp. 2205–2215.

[28] Z. Guo, Y. Zhang, and W. Lu, "Attention guided graph convolutional networks for relation extraction," in *Proc. ACL*, 2019, pp. 241–251.

[29] J. Bastings, I. Titov, W. Aziz, D. Marcheggiani, and K. Simaan, "Graph convolutional encoders for syntax-aware neural machine translation," in *Proc. EMNLP*, 2017, pp. 1957–1967.

[30] C. Shang, Y. Tang, J. Huang, J. Bi, X. He, and B. Zhou, "End-to-end structure-aware convolutional networks for knowledge base completion," in *Proc. AAAI*, vol. 2019, pp. 3060–3067.

[31] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*. [Online]. Available: http://arxiv.org/abs/1312.6203

[32] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," 2015, *arXiv:1506.05163*. [Online]. Available: http://arxiv.org/abs/1506.05163

[33] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*. [Online]. Available: http://arxiv.org/abs/1710.10903

[34] S. Liu, Y. Chen, K. Liu, and J. Zhao, "Exploiting argument information to improve event detection via supervised attention mechanisms," in *Proc. ACL*, 2017, pp. 1789–1798.

[35] Y. Chen, H. Yang, K. Liu, J. Zhao, and Y. Jia, "Collective event detection via a hierarchical and bias tagging networks with gated multi-level attention mechanisms," in *Proc. EMNLP*, 2018, pp. 1267–1276.

[36] Y. Wu *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," 2016, *arXiv:1609.08144*. [Online]. Available: http://arxiv.org/abs/1609.08144

[37] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. AISTATS*, 2011, pp. 315–323.

[38] X. Wang, X. Han, Z. Liu, M. Sun, and P. Li, "Adversarial training for weakly supervised event detection," in *Proc. NAACL-HLT*, 2019, pp. 998–1008.

[39] S. Liu, K. Liu, S. He, and J. Zhao, "A probabilistic soft logic based approach to exploiting latent and global information in event classification," in *Proc. AAAI*, 2016, pp. 2993–2999.

[40] Y. Lu, H. Lin, X. Han, and L. Sun, "Distilling discrimination and generalization knowledge for event detection via delta-representation learning," in *Proc. ACL*, 2019, pp. 4366–4376.

[41] B. Yang and T. M. Mitchell, "Joint extraction of events and entities within a document context," in *Proc. NAACL-HLT*, 2016, pp. 289–299.

[42] T. Zhang, H. Ji, and A. Sil, "Joint entity and event extraction with generative adversarial imitation learning," *Data Intell.*, vol. 1, no. 2, pp. 99–120, May 2019.

[43] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012, *arXiv:1212.5701*. [Online]. Available: http://arxiv.org/abs/1212.5701

[44] S. Liao and R. Grishman, "Using document level cross-event inference to improve event extraction," in *Proc. ACL*, 2010, pp. 789–797.

[45] N. T. Ngo, T. N. Nguyen, and T. H. Nguyen, "Learning to select important context words for event detection," in *Proc. PAKDD*. Cham, Switzerland: Springer, 2020, pp. 756–768.

[46] J. Liu, Y. Chen, K. Liu, and J. Zhao, "Event detection via gated multilingual attention mechanism," in *Proc. AAAI*, 2018, pp. 1–8.

[47] M. Tong, B. Xu, S. Wang, Y. Cao, L. Hou, J. Li, and J. Xie, "Improving event detection via open-domain trigger knowledge," in *Proc. ACL*, 2020, pp. 5887–5897.

[48] S. Cui, B. Yu, T. Liu, Z. Zhang, X. Wang, and J. Shi, "Event detection with relation-aware graph convolutional neural networks," 2020, *arXiv:2002.10757*. [Online]. Available: http://arxiv.org/abs/2002.10757

LEI LI received the B.S. degree in electronic information engineering from Sichuan University, Chengdu, China, in 2018. He is currently pursuing the M.A.Eng. degree with the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing, China. His research interests include deep learning, natural language processing, and information extraction.

LI JIN received the B.S. degree from Xidian University, Xi'an, China, in 2012, and the Ph.D. degree from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, in 2017. He is currently a Research Assistant with the Aerospace Information Research Institute, Chinese Academy of Sciences. His research interests include machine learning, knowledge graph, and geographic information processing.

ZEQUN ZHANG received the B.Sc. and Ph.D. degrees from Peking University, Beijing, China, in 2012 and 2017, respectively. He is currently a Research Assistant with the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing, China. His research interests include information fusion, inverse problems, and natural language processing.

QING LIU received the B.S. degree from the Wuhan University of Technology, Wuhan, China, in 2016, and the M.D. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2019. He is currently an Intern Researcher with the Institute of Electronics, Chinese Academy of Sciences. His research interests include machine learning, knowledge graph, and nature language processing.

XIAN SUN received the B.Sc. degree from Beihang University, Beijing, China, in 2004, and the M.Sc. and Ph.D. degrees from the Institute of Electronics, Chinese Academy of Sciences, Beijing, in 2009. He is currently a Professor with the Aerospace Information Research Institute, Chinese Academy of Sciences. His research interests include computer vision and remote sensing image understanding.

HONGQI WANG received the B.Sc. degree from the Changchun University of Science and Technology, Changchun, China, in 1983, the M.Sc. degree from the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun, in 1988, and the Ph.D. degree from the Institute of Electronics, Chinese Academy of Sciences, Beijing, China, in 1994.

He is currently a Professor with the Aerospace Information Research Institute, Chinese Academy of Sciences. His research interests include computer vision and remote sensing image understanding.