Reconstructing Event Regions for Event Extraction via Graph Attention Networks

Pei Chen[1]*, **Hang Yang**[2,3], **Kang Liu**[2,3],
Ruihong Huang[1], **Yubo Chen**[2,3], **Taifeng Wang**[4], and **Jun Zhao**[2,3]
[1] Texas A&M University, College Station, TX
[2] Institute of Automation, Chinese Academy of Sciences, Beijing, China
[3] University of Chinese Academy of Sciences, Beijing, China
[4] Ant Group, Hangzhou, China
{chenpei, huangrh}@tamu.edu, taifeng.wang@antgroup.com
{hang.yang, kliu, yubo.chen, jzhao}@nlpr.ia.ac.cn

Abstract

Event information is usually scattered across multiple sentences within a document. The local sentence-level event extractors often yield many noisy event role filler extractions in the absence of a broader view of the document-level context. Filtering spurious extractions and aggregating event information in a document remains a challenging problem. Following the observation that a document has several relevant event regions densely populated with event role fillers, we build graphs with candidate role filler extractions enriched by sentential embeddings as nodes, and use graph attention networks to identify event regions in a document and aggregate event information. We characterize edges between candidate extractions in a graph into rich vector representations to facilitate event region identification. The experimental results on two datasets of two languages show that our approach yields new state-of-the-art performance for the challenging event extraction task.

1 Introduction

Event Extraction (EE), a challenging task in Natural Language Processing, aims to extract key types of information (aka *event roles*, e.g., *perpetrators* and *victims* of an *attack* event) that can represent an event in texts and plays a critical role in downstream applications such as Question Answer (Yang et al., 2003) and Summarizing (Filatova and Hatzivassiloglou, 2004). Existing research on EE mostly focused on sentence-level, such as the evaluation in Automatic Content Extraction (ACE) 2005[1]. However, an event is usually described in multiple sentences in a document. As illustrated in Figure 1, relevant event information (noun phrases in green color) is scattered across the whole document. To extract event information accurately and comprehensively at document-level, it is necessary to understand the wider context spanning over multiple sentences.

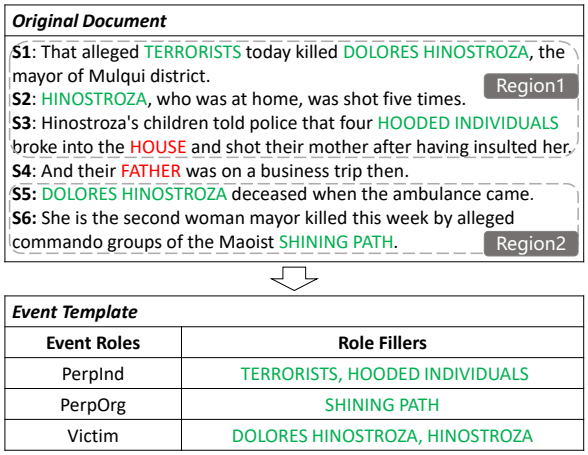

Figure 1: An example of document-level event extraction. We need to extract noun phrases from the document as *role fillers* for the *event roles* in the predefined event template. The uppercased noun phrases in the document are role fillers extracted by the sentence-level extractor. Red phrases are correct while green phrases are noises compared to the standard in the template. There are two *event regions* in the sample document.

The existing approaches for event extraction (EE) often decompose the document-level EE into sentence-level EE, and extract candidate event role fillers from individual sentences one by one. The event role filler extractors often use extraction patterns (Riloff, 1996) or classifiers (Boros et al., 2014) to identify typical local contexts containing a certain type of event role fillers. However, local event role filler extractors often produce many false

*Most of the work was done when the first author was a research engineer in the Institute of Automation, CAS.

[1] http://projects.ldc.upenn.edu/ace/

Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, pages 811–820
December 4 - 7, 2020.

candidates, e.g., the red noun phrases shown in the example document of Figure 1.

As shown in the example, one document often mentions a target event multiple times and each time it takes one or more sentences to articulate the event. The target event role fillers tend to be mentioned in several groups of adjacent sentences, and we define those adjacent relevant sentences as different ***event regions***. For example, in Figure 1, the document mentions the target event twice in two regions. The correct role fillers are crowding in the first event region *S1*, *S2*, *S3* and the second one *S5*, *S6* respectively. Nevertheless, the sentence-level extractor will extract noise from both the event regions like *HOUSE* from *S3* and irrelevant sentence like *FATHER* in *S4*, destroying the layout of the original regions.

Many previous efforts try to avoid aggregating the noisy candidates by detecting such event regions. The popular approach is to apply sentential classification to filter the sentences and recognize role fillers from the chosen sentences (Patwardhan and Riloff, 2009; Huang and Riloff, 2012). However, these approaches only detect regions at single sentence-level and ignore the crowding of relevant sentences. Also, they also suffer from the accumulative error of sentential classification. For example, they may identify *S2* as a relevant event region but *S3* as irrelevant because they fail to take into account the similarity of *S2* and *S3*. Another solution proposed by Yang *et al.* (2018) tries to detect the primary event description sentence and supplement the missing event roles with fillers from adjacent sentences. This method considers the multiple sentences in an event region but is limited to one region per document. For instance, it may detect *S1* as the primary sentence and supplement it with *S2*, missing the valid items like *SHINING PATH* from region 2. Moreover, it also suffers from the errors selecting primary sentence, and the supplementing strategy is coarse-grained and fails to take into account every candidate filler individually.

We build a graph for each document to directly model the multiple event regions in a document, each region potentially consisting of multiple sentences. In each document graph, the nodes are candidate event role fillers and we insert an edge between two nodes based on either positional proximity (in adjacent sentences or within the same sentence) or the coreference relation between two candidate extractions. The document graphs capture sentence similarities and sophisticated discourse connections among the candidate event role fillers to reconstruct the original event regions, which can recognize false event role filler extractions from irrelevant sentences. For example, after identifying the differences between *S4* and adjacent sentences *S3* and *S5*, our model will filter the noisy candidate *FATHER* in *S4*.

Furthermore, constructing document graphs formed by candidate event role fillers and applying graph neural networks will enable recognizing false event role filler extractions within an event region. We employ attentional networks on the graphs to reinforce each candidate's representations by global contextual information and then classify the candidates in a fine-grained manner. Specifically, we characterize the edges into vector representations with rich features to control the information flowing between any two nodes. For instance, this mechanism will be likely to recognize that it is a murder event based on the sentential contexts of sentences *S2* and *S3*, and therefore determine that the candidate extraction *HOUSE* is a false extraction because the Targets of a murder are individuals most commonly, but not physical targets or buildings.

We evaluate our approach on two document-level event extraction datasets: the MUC-4 dataset and a newly created dataset CFEED[2]. Experimental results show that the proposed approach successfully reconstructs 70% of the event regions and yields new state-of-the-art performance for event extraction on both datasets. In summary, the main contributions of this paper are as follows:

- We propose graphs directly modeling the multiple regions with multiple sentences, which successfully help to reconstruct event regions naturally avoid redundant extractions irrelevant sources.
- We propose an edge-enriched graph attention algorithm that can blend both the local clues and global context to enforce semantic representations for each candidate and help to filter noises in the event regions.
- Experimental results show that our method outperforms the existing state-of-the-arts on two datasets with different languages, including a public English MUC-4 dataset and a large-scale Chinese CFEED dataset.

[2] http://www.nlpr.ia.ac.cn/cip/~liukang/dataset/documentevent1.html

2 Related Work

Sentence-level EE has achieved a lot of advancement in recent work (Chen et al., 2015; Nguyen et al., 2016; Chen et al., 2018) and can be classified into template-based approaches (Jungermann and Morik, 2008; Bjorne et al., 2010; Hogenboom et al., 2016) and statistical approaches. Template-based methods require human-crafted templates to match the events. Most of the statistical methods are supervised and either based on feature engineering (Ahn, 2006; Ji and Grishman, 2008; Liao and Grishman, 2010; Reichart and Barzilay, 2012) or Neural network algorithm (Chen et al., 2015; Nguyen et al., 2016; Chen et al., 2018; Liu et al., 2018; Sha et al., 2018; Liu et al., 2018). However, these supervised methods rely on intensive manual annotations. To alleviate this problem, many weak supervised methods (Chen et al., 2017; Zeng et al., 2018) have arisen and achieved good performance in ACE 2005 evaluation.

However, most of the time, people care about the events discussed across a whole document. So research on document-level EE also prevails. Traditionally, pattern-based and classifier-based methods are popular to solve this task. Systems like AutoSlog (Riloff et al., 1993) and AutoSlog-TS (Riloff, 1996) directly applied regular patterns to extract role fillers. Many works (Patwardhan and Riloff, 2007, 2009; Huang and Riloff, 2011, 2012; Boros et al., 2014) relied on feature-based classifiers to distinguish candidate role fillers from texts and achieved better performance. Until recent years, researchers (Hsi, 2018; Yang et al., 2018; Zheng et al., 2019) began to utilize multiple neural-based methods to solve the task. Notably, among the document-level EE research, some works (Patwardhan and Riloff, 2009; Huang and Riloff, 2012; Yang et al., 2018) have noticed the importance of identifying event regions to improve performance.

Traditional neural networks such as Convolutional Neural Networks and Recursive Neural Networks are hard to deal with graphical data structures, so many graph-based neural networks (GNNs) emerge (Gori et al., 2005; Bruna et al., 2013; Kipf and Welling, 2016). In order to deal with graphs with different edge types, relational GNNs (Schlichtkrull et al., 2018; Marcheggiani and Titov, 2017; Vashishth et al., 2019; Bastings et al., 2017) try to use separate weights for different edges. However, one limitation of these GNNs is that the weights are fixed for all neighbors. So Veličković et al. (2017) leveraged masked attentional layers (GATs) to learn adaptive weights for different neighbors. By now, some works (Schlichtkrull et al., 2018; Vashishth et al., 2019) have successfully applied GNNs to model the document-level information within texts and achieved state-of-the-art performance. Our model is distinguishing because we not only utilize these recent advances but also turns the relational edges to feature-enriched nodes and extends GATs on such heterogeneous graphs.

3 Fine-grained Filtering Framework

3.1 Overall Framework

Our method for document-level Event Extraction follows three main procedures.

Extracting role candidates by sentence-level event extractor (SEE): Given a document, we disintegrate it into a series of sentences and apply sentence-level event extractors to identify candidate role fillers.

Constructing graphs to model event regions: Based on the primitive results from the last step and the properties of event regions, we build graphs to capture both the local clues and global context among those candidates.

Selecting role fillers via edge-enriched graph attention networks (EE-GAT): We encode the different edges into vectors and then leverage the attention mechanism on the edge-enriched graphs to update the nodes' representations. After that, we feed the candidates to classifiers for filtering.

3.2 Extracting Role Candidates by Sentence-level Event Extractor

Sentence-level Event Extractor aims at extracting event roles from each sentence in a document. We reproduce the SEE introduced by Yang *et al.* (2018) and employ BiLSTM-CRF to identify candidates from each sentence. The model uses the word embedding as the input features, and this method is compatible with both the English and Chinese corpus.

3.3 Constructing Graphs to Model Event Regions

For each document, we want to utilize the observed event region information in our model. As discussed before, the original event region information of the candidates from the SEE is destroyed. So we make use of the properties of the original

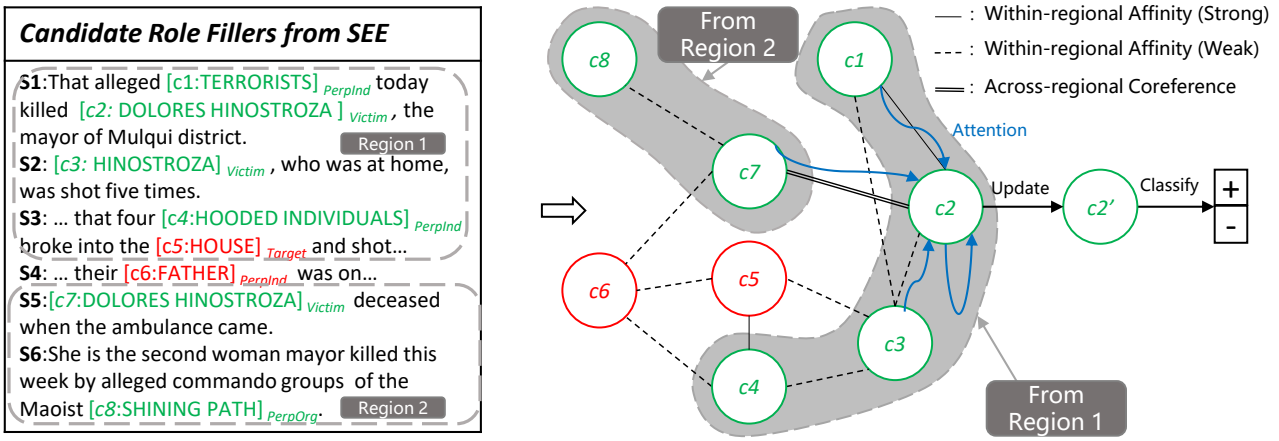

Figure 2: The overall framework of fine-grained filtering framework. 8 candidate role fillers ($c1 - c8$) with sentential clues and specific role types are extracted by SEE as nodes. 3 types of edges are defined to connect those nodes: within-regional affinity (Strong), within-regional affinity (weak), across-regional coreference. Then we employ edge-enriched attention mechanism to update the representation of each candidate for classification, like node $c2'$ from $c2$. Ideally, the framework will filter noisy candidates $c5$, $c6$ and reconstruct the original two event regions.

event regions and, according to them, build a graph to link those candidates. Specifically, we first take each candidate role filler as the node in the graph. These nodes can easily take rich candidates' rich features as initial representation, such as the entity embeddings and the local sentential information. For example, in Figure 2, we extract 8 candidate role fillers with specific role type from a document using the aforementioned SEE. We mark them as $c1 - c8$ and regard them as the nodes.

As we know from the property of event regions, the correct role fillers tend to crowd within the same or adjacent sentences, such as $c1$, $c2$, $c3$ and $c4$ in Figure 2. Also, one event may be mentioned by multiple event regions, and there can be coreferential role filler across these regions, like $c2$ and $c7$. We employ such properties of event regions to construct the graphs so as to utilize regional information. In detail, we define the following 2 types of relations (3 types of edges) in the graphs:

Within-regional Affinity When two candidates appear in the same or adjacent sentences, they have a within-regional affinity. We use such affinities to model the phenomenon that multiple event role fillers tend to crowd in an event region. When one candidate filler in the region has high confidence to be a positive one, other candidates can share this confidence and vice versa. Furthermore, we distinguish the same sentence affinity from the adjacent sentences affinity using different edges because we believe such affinity is stronger within the same sentence. For instance, in Figure 2, we assign $c1$ and $c2$ with strong within-regional affinity since they are both in *S1*, and use a single solid line to represent this affinity. And we assign $c6$ and $c7$ with the weak within-regional affinity because they occur in adjacent sentences *S4* and *S5* respectively. A single dotted line is used to illustrate it. The weak affinity may have less confidence sharing and help filter nosy candidate $c6$ while keeping $c7$.

Across-regional Coreference When two candidates are the same to each other lexically and also recognized as the same event role type, we assume that they have a coreference relationship. When these two coreferential candidates are not in the same or adjacent sentences (they do not have within-regional affinity), we assign them with across-regional coreference so as to bridge different regions. This is because a document usually mentions the target event in multiple event regions, and the same event role fillers may repeat in these regions. We connect these regions by utilizing such cross-region coreference relationships. Such connections will help exchange semantic information and share classification confidence among different regions. Here in Figure 2, we assign $c2$ and $c7$ with across-regional coreference relationship and use a double solid line to represent corresponding edge in the graph.

Although the constructed graphs do not precisely demonstrate the original event regions, the GNNs models will synthesize comprehensive context from such connections to enforce each candidate's representations, identify the noises, and reconstruct the original regions as a result.

3.4 Selecting Role Fillers via Edge-enriched Graph Attention Networks

After building graphs from the documents, we classify the nodes via supervised learning. We first encode the nodes and edges into vectors and then apply the attention mechanism to update the representation of each node from its neighbors, and finally feed the updated representation into classifiers for filtering.

Encoding Each graph is represented by its nodes and edges, as $G = (C, E)$, where C represents nodes and E represents edges. We first initialize all nodes with their feature representations and get $C = \{c_1, c_2, \ldots, c_n\}, c_i \in R^F$, where c_i represents the features of node i, n is the number of nodes and F is the embedding size for each node. Each node is featured by 4 types of embeddings $c_i = [w_i, p_i, t_i, s_i]$, where w_i is the average word embedding of each candidate entity, p_i is the position embedding of the candidate with respect to the sentence, t_i is the embedding of role type, and s_i is the sentence embedding by averaging all words in the sentence.

For edges, the plain graph attention mechanism does not encode them into vectors. Such a mechanism equally treating the edges suffers from losing the information of distinguishing edges. A popular way to deal with this problem is to use different weights for different edges in the attention operation (Relational GAT, R-GAT). However, R-GAT does not have edge representation nor controls the information flow equally for the same type edges. Our edge-enriched attention model characterizes the edges into vector representations, which can especially control the information between each candidate node pair. Initially, we regard each edge as a new type of node featuring its edge type and make a new set of nodes $E^{'}$. For example in Figure 3, we use the new node $e_{1,2} \in E^{'}$ to represents the original within-regional affinity edge between nodes c_1 and c_2. Here the same type of edges will share the same initial vector representation.

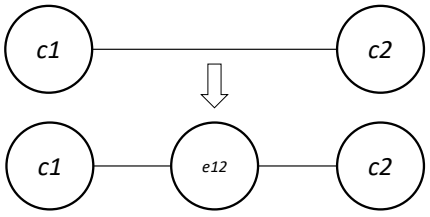

Figure 3: Encoding of Edges

In this way, we construct a new graph $\widetilde{G} = (\widetilde{C}, \widetilde{E})$ in which all the new edges in the graph are the same, but we have two types of nodes $\widetilde{C} = \{C, E^{'}\}$, which means the graph is heterogeneous now. To update all nodes in the same attention mechanism, we combine the feature spaces of both the original nodes and new edge-enriched nodes. In this way, any new node within the new graph will have 5 types of embedding: $\tilde{c}_i = [w_i, p_i, t_i, s_i, e_i]$, where $[e_i]$ is the edge type representation. We initialize e_i as zero vectors for original candidate nodes and the other 4 embeddings as zero vectors for the new edge nodes.

Updating Then we update the edge-enriched graph based on GAT proposed by (Veličković et al., 2017). GAT is in essence masked attention operation on graphs. For each layer of graph attention, it updates the representation of node $\tilde{c}_i$ by computing the linear combinations of its neighbors' normalized attention scores and their corresponding transformed representations:

$$\tilde{c}_i^{'} = \Big\Vert_{h=1}^{H} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^h W^h \tilde{c}_j\right) \quad (1)$$

Here we concatenate (signified by $\|$) H heads of the attentions results. σ represents the activation functions and $\mathcal{N}_i$ represents the neighbor nodes of $\tilde{c}_i$, including itself. Transformation W^h is shared for all nodes within each head. We obtain the attention score α_{ij}^h in head h as followed:

$$\alpha_{ij}^h = \frac{\exp\left(\text{LeakyReLU}\ \left(a^T\left(W^h\tilde{c}_i \| W^h\tilde{c}_j\right)\right)\right)}{\sum_{k \in N(i)} \exp\left(\text{LeakyReLU}\ \left(a^T\left(W^h\tilde{c}_i \| W^h\tilde{c}_k\right)\right)\right)} \quad (2)$$

Here a is a single-layer feedforward neural network. We apply two layers of the GAT to update on the graphs. The first layer will exchange the information between candidate nodes and edge nodes, which will characterize the edge representation with the semantic context. Now each edge node will have unique vector representations. Then in the second layer, the candidate nodes will incorporate information from the updated edge nodes, indirectly blend in the features of adjacent candidate nodes in the original graph G. The enriched edges play the role to control the information flowing between neighbor candidate nodes uniquely.

For comparison, the R-GAT model uses different weights for different edges as followed, where $\mathcal{R}$ is the set of edge types. Here different edges control

Systems	Event Roles in MUC-4 Dataset					
	PerpInd	**PerpOrg**	**Target**	**Victim**	**Weapon**	**Average**
(Riloff, 1996)	33/49/40	53/33/41	54/59/56	49/54/51	38/44/41	45/48/46
(Patwardhan and Riloff, 2009)	51/58/54	34/45/38	43/72/53	55/58/56	57/53/55	48/57/52
(Huang and Riloff, 2011)	48/57/52	46/53/50	51/73/60	56/60/58	53/64/58	51/62/56
(Huang and Riloff, 2012)	54/57/56	55/49/51	55/68/61	**63**/59/61	62/64/63	58/60/59
(Boros et al., 2014)	53/58/55	56/67/**61**	59/63/61	56/55/55	72/65/68	59/61/60
(Yang et al., 2018)	48/60/54	52/74/**61**	52/70/59	56/62/59	70/77/74	56/69/61
SEE	35/**77**/48	28/**88**/42	44/**80**/57	38/**83**/53	59/**86**/70	41/**83**/55
GAT	**62**/52/57	57/53/55	60/61/60	61/58/59	**78**/78/**78**	**64**/60/62
R-GAT	58/62/**60**	57/61/59	60/63/62	57/67/61	71/75/73	61/66/63
EE-GAT	60/59/**60**	**58**/61/60	**61**/68/**64**	62/65/**63**	75/75/75	63/66/**65**

Table 1: Evaluation on MUC-4 test set, P/R/F1 (Precision/Recall/F1-Score,%).

Event Types	Systems	Event Roles in CFEED Dataset					
		NAME	**NUM**	**BEG**	**END**	**ORG**	**Average**
Freeze	(Boros et al., 2014)	71/76/74	56/57/56	**77**/54/63	83/**80**/**81**	70/80/**75**	72/69/70
	(Yang et al., 2018)	**83**/71/**76**	**70**/49/58	75/67/71	**85**/65/74	**71**/67/69	**77**/64/70
	EE-GAT	68/**82**/75	57/**63**/**60**	71/**77**/**74**	84/79/**81**	65/**82**/72	69/**77**/**73**
Pledge	(Boros et al., 2014)	74/**95**/83	60/46/52	68/**81**/74	74/**30**/42	83/**92**/87	72/69/70
	(Yang et al., 2018)	**84**/87/**86**	76/54/63	**81**/72/76	**85**/28/42	**88**/82/85	**83**/64/72
	EE-GAT	77/**95**/85	**79**/**55**/**65**	76/78/**77**	83/**30**/**44**	84/91/**88**	80/**70**/**75**
OW/UW	(Boros et al., 2014)	49/**89**/63	63/**65**/64	39/**79**/52	62/**45**/53	—	54/**70**/61
	(Yang et al., 2018)	**77**/70/**73**	79/54/64	66/68/67	74/39/51	—	**74**/58/65
	EE-GAT	66/82/**73**	**80**/60/**68**	**73**/**79**/**76**	**77**/44/**56**	—	**74**/66/**70**
Total	(Boros et al., 2014)	65/**87**/74	60/56/58	61/71/66	73/**52**/61	77/86/**81**	66/69/67
	(Yang et al., 2018)	**81**/76/**78**	**75**/52/61	**74**/69/71	**81**/44/57	**80**/75/77	**78**/62/69
	EE-GAT	70/86/77	72/**59**/**65**	73/**78**/**75**	**81**/51/**63**	75/**87**/**81**	74/**71**/**72**

Table 2: Evaluation on the CFEED test set, P/R/F1 (Precision/Recall/F1-Score,%).

the information exchange differently. However, this mechanism is not as effective as the enriched edges in our EE-GAT model.

$$c_i^{'} = \Big\Vert_{h=1}^{H} \sigma\left(\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}_i} \alpha_{ij}^{h} W^{r,h} c_j\right) \quad (3)$$

Classification After updating the candidate nodes via the two layers multi-head attention mechanism, we need to classify each candidate node as either positive or negative. Now we average the vectors of multiple heads to get the final representation of each node and then project the results into a softmax classification layer.

As a result, we will get the probabilities of the node as either positive or negative. This process is illustrated in equation (4), where $y_i \in \{0, 1\}$ is the label of node i, θ represents all the parameters, p is the probability of y_i equals to 0 or 1.

$$p(y_i|\widetilde{G};\theta) = \text{softmax}\left(\frac{1}{H}\sum_{h=1}\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{h} W^{h} \tilde{c}_j^{'}\right) \quad (4)$$

We train our model to minimize the cross-entropy loss in the data and use the Adam optimization method proposed by Kingma and Ba (2014) to update the parameters θ. The loss function is as followed in equation (4) where $\hat{y_i} = p(y_i = 1|G;\theta)$ is the predicted probability of node i as positive, N is the number of samples.

$$L(\theta) = -\sum_{i=1}^{N}(y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i)) \quad (5)$$

4 Experiments

4.1 MUC-4

MUC-4 dataset was released by Message Understanding Conferences in 1992. It is about terrorism events and consists of 1700 documents as in Table 4. We follow the same evaluation paradigm as previous work and evaluate the 5 kinds of event roles: *PerpInd*, (individual perpetrator), *PerpOrg* (organizational perpetrator), *Target* (physical target), *Victim* (human target name or description)

Datasets	Event Types	Train	Dev	Test	Total
MUC-4	Terrorism	1300	200	200	1700
CFEED	Freeze	589	150	300	1039
	Pledge	3602	300	300	4202
	OW/UW	1303	300	300	1903

Table 3: Statistics of MUC-4 and CFEED

and *Weapon* (instrument id or type). We use head noun matching (e.g. *HINOSTROZA* is considered to match *DOLORES HINOSTROZA*) as before too.

Baselines For comparison, we choose the following 6 previous state-of-the-art systems as the baselines for MUC-4.

Riloff (1996) automatically produced many domain-specific extraction patterns for role fillers extraction.

Patwardhan and Riloff (2009) incorporated both phrasal and sentential evidence to label role fillers. They first used a sentential event recognizer to select sentences and then applied a plausible role-filler recognizer to extract role fillers.

Huang and Riloff (2011) designed TIER system to better extract role fillers from Secondary Context, regardless of whether a relevant event is mentioned.

Huang and Riloff (2012) defined many features and used SVMs to extract local candidate role fillers and CRF to choose sentences for final results.

Boros et al. (2014) utilized domain-relevant word representations as the features of noun phrases and then applied randomized decision trees to identify role fillers. Here we adopt the same idea but use a different classifier MLP. Besides, we use the same node features as in EE-GAT instead of just domain word vectors for comparison with our model.

Yang et al. (2018) proposed a document-level EE system following three steps. It first extracted candidate role fillers from each sentence via sequence tagging model; then it applied Convolutional Neural Networks to detect the primary sentence that mentions the target event; finally, it aggregated the candidate role fillers from the primary sentence and supplements the missing even roles from adjacent sentences.

Experiments on MUC-4 For node representations, we randomly initialize p_i, t_i as 50-dim vectors and e_i as 200-dim, and use the 100-dim Glove[3] word embedding for w_i, s_i. Each layer of the attention mechanism has 8 heads and the learning rate is set as 5e-4. We train on MUC-4 training data for 100 epochs and choose the best model performed on the development set for testing.

We report Precision/Recall/F1-score of the test results for each event role individually and the macro-average over all five roles. The test results are shown in Table 2. From the table, we have the following observations: (1) In general, our EE-GAT framework achieves the best performance compared with previous state-of-the-art methods. It significantly improves the previous best method by 4.0% (65% vs. 61%) on average F1 score and most of the improvement is contributed by the better precision 7.0% (63% vs. 56%) as opposed to Yang et al. (2018). (2) The SEE results have high recall but very low precision because of the noisy candidates. Plain GAT filters some noises and improves precision a lot. R-GAT and EE-GAT balance the trade-off between precision and recall and achieve a better overall F1 score. (3) In detail, our method achieves the best performance nearly on most of the event roles. We significantly improve the F1 score of 4.0% (60% vs. 56%) in *PerInd* and 3.0% in *Target* (64% vs. 61%) compared to previous best in Huang and Riloff (2012).

4.2 CFEED

CFEED **C**hinese **F**inancial **E**vent **E**xtraction **D**ataset is a larger dataset in Chinese about the major events in the announcements of listed companies. We construct it by the same method proposed by Yang *et al.* (2018). We crawled the public announcements from sohu.com[4] and the event templates from eastmoney.com[5], and then align them. We assume that if the key role fillers in a template appear in an announcement, the announcement is describing the event in the template. As in Table 3, it consists of a total of 7144 documents and 3 types of financial events: freezing shares (*freeze*), pledging shares (*pledge*) and overweighting and underweighting shares (*OW&UW*). We defined 5 types of event role in these financial events: shareholder's name (*NAME*), organization (*ORG*), number of shares (*NUM*), event starting date (*BEG*), event ending date (*END*). Note that the *ORG* is not included in *OW&UW* event.

Baselines For comparison, we select the two methods mentioned above as the baselines for CFEED: **Boros et al. (2014)** and **Yang et al. (2018)**.

Experiments on CFEED We use the same settings as in MUC-4 to evaluate on the CFEED except that we use the character-level 100-dim embeddings trained on Chinese wiki corpus[6]. We sep-

[3] https://nlp.stanford.edu/projects/glove/

[4] http://q.stock.sohu.com/index.shtml

[5] http://choice.eastmoney.com/

[6] https://github.com/Embedding/Chinese-Word-Vectors

Statistics	MUC-4			CFEED		
	Gold	SEE	EE-GAT	Gold	SEE	EE-GAT
Avg #Fillers /Doc	8.21	11.17	6.30	11.72	29.95	10.43
Avg #Regions /Doc	1.76	2.86	1.57	2.53	2.21	2.58
Avg #Fillers /Region	5.32	5.54	4.57	5.88	16.94	5.51
Eval for Regions	—	21/87/34	65/70/68	—	16/96/27	68/77/72

Table 4: Distributions of role fillers in the golden data and results of SEE and EE-GAT on the test set of MUC-4 and CFEED. The last row is the evaluation (Precision/Recall/F1-Score,%) of the regions sentence by sentence. The statistics demonstrate the salient Event Regions in golden data and its reconstruction by EE-GAT.

Settings	MUC-4	CFEED			
		Freeze	Pledge	OW&UW	Total
(Yang et al., 2018)	56/**69**/61	**77**/64/70	**83**/64/72	**74**/58/65	**78**/62/69
EE-GAT w/ 1st Rel	**63**/59/61	71/72/71	77/68/72	64/68/66	71/69/70
EE-GAT w/ 1st & 2nd Rels	62/64/63	66/**77**/71	76/**71**/73	64/**70**/67	69/**73**/71
EE-GAT	**63**/66/**65**	69/**77**/**73**	80/70/**75**	**74**/66/**70**	74/71/**72**

Table 5: Effectiveness of the Regional Relations in EE-GAT (Average P/R/F1, Precision/Recall/F1-Score,%). *1st Rel* means strong within-regional affinity and *2nd Rel* means weak within-regional affinity.

arately evaluate the 3 types of events and the results are in Table 3. We can observe that our EE-GAT can achieve the best performance on all the 3 types of events when compared with the baselines. The results verify the robustness of our method in Chinese corpus. Besides, compared with the method in Yang *et al.* (2018), the major improvement comes from recall rather than precision as on MUC-4. This is because the financial announcement documents in CFEED usually have one main sentence describing the target event, so Yang's method can achieve high precision by detecting the primary event mention. However, MUC-4 dataset does not have such characteristics.

4.3 Reconstructing Event Regions

As in Table 4 about event regions, test if a sentence in the new regions appears in the golden regions and get the evaluation *Precision*, *Recall*, and *F1* scores. We can observe that in both of the datasets: (1) EE-GAT successfully reconstructs 70% of the event regions during the evaluation, which improves about 40% from the SEE results. The detection of the event regions contributes to most of the filtering process. (2) SEE extracted too many noisy role fillers compared to the golden standard. EE-GAT filters many noises and the counts of remaining fillers are similar to the golden standard. (3) The distribution of role fillers and event regions are more close to the golden standard after EE-GAT filtering. In detail, on the gold test sets, there are about 1.76 regions in a document and 5.32 fillers in each region on MUC-4, and 2.53 regions and 5.88 fillers per region on CFEED. However, the event region distribution diverges after SEE because of the noisy candidates, and we have about 2.86 regions in a document and 5.54 fillers in each region on MUC-4, and 2.21 regions and 16.94 fillers per region on CFEED. Then these statistics recover back to normal after the filtering of EE-GAT, and there are about 1.57 regions in a document and 4.57 fillers in each region on MUC-4, and 2.58 regions and 5.51 fillers per region on CFEED.

4.4 Effectiveness of Regional Relations

We set the following control experiments to demonstrate the effectiveness of the regional relations in filtering the noise. We add the three types of edges one by one and test the performance of EE-GAT. As in Table 5, we can observe that the overall performance on all the datasets improves when more types of relations are used. (1) Particularly, even the utilization of strong within-regional affinity (1st Rel) only in EE-GAT achieves slightly better performance compared to the previous state-of-the-art (Yang et al., 2018). (2) Adding the weak within-regional affinity (2nd Rel) further improves the overall performance, especially the average 4.5pp improvement in recall score. (3) And the complete EE-GAT model connecting the multiple event regions achieves even better overall performance. These results demonstrate that the event region relations can capture the global contextual information and help to filter the noisy candidates.

5 Conclusion

We propose a fine-grained filtering framework to address the aggregating problem in document-level

event extraction by reconstructing event regions. Our method can filter those noise both in irrelevant sentences and in the event regions and achieve state-of-the-art performance on both the MUC-4 and CFEED datasets. Future work may consider using an end2end model to avoid error propagation from SEE.

Acknowledgments

This work is supported by the Natural Key RD Program of China (No.2018YFB1005100), the National Natural Science Foundation of China (No.61922085, No.U1936207, No.61806201) and the Key Research Program of the Chinese Academy of Sciences (Grant NO. ZDBS-SSW-JSC006). This work is also supported by CCF-Tencent Open Research Fund, Beijing Academy of Artificial Intelligence (BAAI2019QN0301) and independent research project of National Laboratory of Pattern Recognition.

References

David Ahn. 2006. The stages of event extraction. In *The Workshop on Annotating and Reasoning about Time and Events*, pages 1–8.

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. *arXiv preprint arXiv:1704.04675*.

Jari Bjorne, Filip Ginter, Sampo Pyysalo, Jun'Ichi Tsujii, and Tapio Salakoski. 2010. Complex event extraction at pubmed scale. *Bioinformatics*, 26(12):i382–i390.

Emanuela Boros, Romaric Besançon, Olivier Ferret, and Brigitte Grau. 2014. Event role extraction using domain-relevant word representations. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1852–1857.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. 2013. Spectral networks and locally connected networks on graphs.

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 ACL*.

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.

Elena Filatova and Vasileios Hatzivassiloglou. 2004. Event-based extractive summarization. *Text Summarization Branches Out*.

Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734. IEEE.

Frederik Hogenboom, Flavius Frasincar, Uzay Kaymak, Franciska De Jong, and Emiel Caron. 2016. A survey of event extraction methods from text for decision support systems. *Decision Support Systems*, 85(C):12–22.

Andrew Hsi. 2018. *Event Extraction for Document-Level Structured Summarization*. Ph.D. thesis, Carnegie Mellon University.

Ruihong Huang and Ellen Riloff. 2011. Peeling back the layers: detecting event role fillers in secondary contexts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1137–1147. Association for Computational Linguistics.

Ruihong Huang and Ellen Riloff. 2012. Modeling textual cohesion for event extraction. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of the ACL*, pages 254–262.

Felix Jungermann and Katharina Morik. 2008. *Enhanced Services for Targeted Information Retrieval by Event Extraction and Data Mining*. Springer Berlin Heidelberg.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of the ACL*, pages 789–797.

Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2018. Event detection via gated multilingual attention mechanism. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826*.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the ACL*, pages 300–309.

Siddharth Patwardhan and Ellen Riloff. 2007. Effective information extraction with semantic affinity patterns and relevant regions. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 717–727.

Siddharth Patwardhan and Ellen Riloff. 2009. A unified model of phrasal and sentential evidence for information extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 151–160. Association for Computational Linguistics.

Roi Reichart and Regina Barzilay. 2012. Multi event extraction guided by global constraints. In *Proceedings of the ACL*, pages 70–79.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proceedings of the national conference on artificial intelligence*, pages 1044–1049.

Ellen Riloff et al. 1993. Automatically constructing a dictionary for information extraction tasks. In *AAAI*, volume 1, pages 2–1. Citeseer.

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 *European Semantic Web Conference*, pages 593–607. Springer.

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Shikhar Vashishth, Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2019. Dating documents using graph convolution networks.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks.

Hang Yang, Yubo Chen, Kang Liu, Yang Xiao, and Jun Zhao. 2018. Dcfee: A document-level chinese financial event extraction system based on automatically labeled training data. In *Proceedings of ACL 2018, System Demonstrations*, pages 50–55.

Hui Yang, Tat-Seng Chua, Shuguang Wang, and Chun-Keat Koh. 2003. Structured use of external knowledge for event-based open domain question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 33–40. ACM.

Ying Zeng, Yansong Feng, Rong Ma, Zheng Wang, Rui Yan, Chongde Shi, and Dongyan Zhao. 2018. Scale up event extraction learning via automatic training data generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Shun Zheng, Wei Cao, Wei Xu, and Jiang Bian. 2019. Doc2EDAG: An end-to-end document-level framework for Chinese financial event extraction. 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 337–346, Hong Kong, China. Association for Computational Linguistics.