Computers, Materials & Continua
DOI:10.32604/cmc.2021.017028
Article

Tech Science Press

Joint Event Extraction Based on Global Event-Type Guidance and Attention Enhancement

Daojian Zeng[1], Jian Tian[2], Ruoyao Peng[1], Jianhua Dai[1,*], Hui Gao[3] and Peng Peng[4]

[1]Hunan Normal University, Changsha, 410081, China
[2]Changsha University of Science & Technology, Changsha, 410114, China
[3]National University of Defense Technology, Changsha, 410073, China
[4]University of Waterloo, Waterloo, N2L3G1, Canada
[*]Corresponding Author: Jiahua Dai. Email: jhdai@hunnu.edu.cn
Received: 18 January 2021; Accepted: 29 March 2021

Abstract: Event extraction is one of the most challenging tasks in information extraction. It is a common phenomenon where multiple events exist in the same sentence. However, extracting multiple events is more difficult than extracting a single event. Existing event extraction methods based on sequence models ignore the interrelated information between events because the sequence is too long. In addition, the current argument extraction relies on the results of syntactic dependency analysis, which is complicated and prone to error transmission. In order to solve the above problems, a joint event extraction method based on global event-type guidance and attention enhancement was proposed in this work. Specifically, for multiple event detection, we propose a global-type guidance method that can detect event types in the candidate sequence in advance to enhance the correlation information between events. For argument extraction, we converted it into a table-filling problem, and proposed a table-filling method of the attention mechanism, that is simple and can enhance the correlation between trigger words and arguments. The experimental results based on the ACE 2005 dataset showed that the proposed method achieved 1.6% improvement in the task of event detection, and obtained state-of-the-art results in the argument extraction task, which proved the effectiveness of the method.

Keywords: Event extraction; event-type guidance; table filling; attention mechanisms

1 Introduction

Event extraction (EE) is an essential yet challenging task for information extraction. It is widely used in natural language processing, especially in the fields of automatic expansion of large-scale knowledge bases, automatic summarization, and biomedicine [1]. Therefore, in recent years, a lot of research has been conducted on event extraction tasks, that aimed to extract trigger words from unstructured natural text, determine the event type of trigger words, and extract the argument related to the event and determine the role played in the event. The ACE 2005

evaluation conference defined event extraction as two subtasks: event detection (identifying and classifying event triggers) and argument extraction (identifying arguments of event triggers and labeling their roles).

The traditional methods generally handle event extraction as a pipeline of two separate tasks: event detection and argument extraction. The pipeline method achieves good results, especially when deep learning techniques are used. The most successful pipelined method was proposed by Chen et al. [2], who used dynamic multiple pooling convolutional neural networks to automatically learn features from sentences and represented words with continuous representations [3–5]. However, as the pipeline method is divided into two subtasks, the interrelationship between the subtasks is ignored. Specifically, the result of event detection affects the following argument extraction, and the effect of argument extraction promotes the result of event detection [6]. Thus, researchers turned to the method of joint extraction.

Li et al. [6] performed one of the most successful studies of the joint method, which is based on a structure-aware algorithm with sets of local and global features for EE. The interdependence between trigger words and arguments is captured by global features. This method alleviates the shortcomings of the pipeline method and achieves good results. However, this feature extraction relies on natural language processing tools (e.g., part of speech tagging) and has poor generalization capabilities for new words and unseen features. Therefore, Nguyen et al. [7] proposed joint event extraction based on the Recurrent Neural Network (RNN). They used recurrent neural networks to automatically learn rich contextual semantic representations. In order to capture the interdependence between trigger words and arguments, memory vectors and matrices are introduced in the method to store prediction information in the process of sentence labeling. To a certain extent, this method solves the deficiencies of Li et al. [6] method, but it does not make full use of the syntactic dependence between the components in the sentence. Sha et al. [8] used a dependency bridge based on a bi-directional RNN to learn the syntactic dependency between each component in a sentence and introduce a tensor to learn the interdependence between arguments. However, all of the above methods have a common disadvantage: They ignore the interdependence of multiple events in the sentence.

In actual event extraction scenarios, there will inevitably be multiple events in one sentence. Compared with single event extraction, it is more complicated to accurately extract multiple events. There is a strong correlation among events drawn from the same sentence. For example, as shown in Fig. 1, the *Attack* event helps us determine that the word ***died*** triggers the *Die* event rather than the *Inject* event. It is worth noting that multiple event phenomena are ubiquitous in natural language. According to statistics, there are 3,978 incident-related sentences in the ACE 2005 dataset, and 1,042 sentences contain multiple events, accounting for 26.6% of the entire incident dataset. It is common for multiple events in the sentence to require extraction. Liu et al. [9] conducted an in-depth study on multiple event extraction. They used the graph convolutional neural to learn the dependency syntax relationship between the components in the sentence, and tried to capture the correlation between events. However, owing to the complexity of the dependency syntax tree and reliance on NLP tools for preprocessing, this method inevitably encounters the error propagation problem and the interdependence between events is not fully resolved.

In order to solve the above problems, we proposed a joint event extraction method based on global event type guidance and the attention enhancement mechanism. Recent studies on multi-task learning (MTL) in deep neural networks for NLP revealed that multi-task gains were more likely for target tasks that quickly plateaued with non-plateauing auxiliary tasks [10]. Because of the compelling benefits of MTL, we proposed a multi-task setup for identifying and classifying

events and arguments. Specifically, we first use the BERT pre-training model to encode the sentence in order to obtain the context information of each token. Next, the event guidance layer is exploited to predict the candidate event types of the input sentence. At the same time, we introduce the CRF layer to identify the candidate arguments. Then, we feed the candidate event types and context features into the softmax layer for trigger word recognition and event classification. Finally, we enumerate the combinations between two tokens in the sentence. The corresponding context features, candidate argument features, trigger words and event-type features are attentively considered for argument role classification in a table-filling [11–13] manner (see Fig. 2). From the above, we noticed that the event types predicted by the event guidance layer helped to guide event classification. With the injected events, we allow the network to be aware of all the events that exist in the sentence in advance. Thus, the interdependencies of events are taken into account. Moreover, we use an attention mechanism to comprehensively take all tokens into account for the role classification of any two tokens. Therefore, the correlation between trigger words and arguments is taken into account at the table filling stage. In summary, the contributions of this work can be summarized as follows:

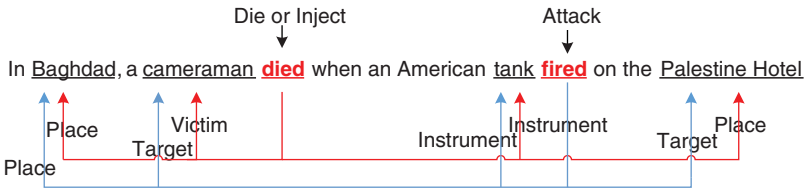

Figure 1: An example of multiple events. There are two events in the sentence: a *Die* event triggered by the word *died*, with four arguments in the red, and an *Attacked* event triggered by the word *fired*, with four arguments in the blue

	In	Baghdad	a	cameraman	died	···	tank	fired	on	Palestine	Hotel
In											
Baghdad											
a											
cameraman											
died		Place		Victim			Instrument			Place	
···											
tank											
fired		Place		Target			Instrument			Target	
on											
Palestine											
Hotel											

Figure 2: The Triggers-Argument role table for the example in Fig. 1. "*Died*" and "*fired*" are two trigger words. *Place, Victim*, *Instrument* and *Target* represent the argument roles. Blank cells indicate there is no argument role

(1) We proposed a novel event-type guidance layer to predict the event types of the input sentence. The candidate event types are used to guide trigger word recognition and event detection, which can strengthen the complex interdependencies of events.

(2) The method converts the argument extraction as a table-filling problem. An attention mechanism is introduced to involve the representations of multiple tokens, which automatically discovers some useful contextual information for argument role classification.
(3) We conducted wide experiments on the ACE 2005 dataset. The experimental results indicate that the proposed method outperforms several strong baselines.

2 Related Work

Traditional event extraction methods usually exploit a pipelined way to extract events where arguments are identified using a classifier after event detection [14,15]. These methods have a fatal flaw: They ignore the underlying interdependencies between event detection and argument extraction and suffer from error propagation.

To address the above problem, a joint event extraction method based on the Markov logic network [16–18] was proposed. Afterward, the structured perceptron [6,19] and the dual decomposition method [20] were successively proposed for event extraction.

Recently, with the widespread application of neural networks in machine translation, text classification, steganography analysis [21,22], and other fields, researchers have also tried to use neural networks to complete event extraction. For example, Chen et al. [2] employed dynamic multi-pooling convolutional neural networks to automatically learn features, in which the input words are represented by pre-trained word embeddings [3–5]. However, they achieved promising results, and the method still follows the pipelined framework. Similarly, Nguyen et al. [7] proposed a joint approach named JRNN, in which recurrent neural networks are used to automatically learn the rich contextual semantic representation of sentences. The relations between event triggers with specific subtypes and their corresponding arguments are captured by the devised memory vectors and matrices. Similarly, Sha et al. [8] exploited dependency bridges to connect syntactically related words based on a bidirectional recurrent neural network. Moreover, a tensor layer was proposed on each pair of two candidate arguments, which enables intensive, argument-level information interaction. Liu et al. [9] conducted an in-depth study on multi-event extraction, which introduced a syntactic dependency tree and used the graph convolutional neural networks to learn the syntactic dependency of each component for the sentence.

The above mentioned joint extraction methods have achieved good results. However, these existing methods have a common disadvantage: They do not consider the situation where multiple events appear in one sentence at the same time. To solve this problem, we proposed an event-guided and attention enhancement joint approach for event extraction. The pre-predicted event-type information allows for better event detection, and the attention mechanism is exploited to leverage the sentential context.

3 Methodology

Our proposed joint model consists of six modules: (i) BERT, (ii) NER, (iii) Event-Types Proposal, (iv) Event Detection (v) Token Pair Attention, and (vi) Table Filling, as illustrated in Fig. 3. Given a sentence as the model input, the model first generates a deep contextualized word representation for each token using BERT. Next, the event guidance layer is exploited to predict the candidate event types of the input sentence. At the same time, we introduce the CRF layer to identify the candidate arguments. Then, we feed the candidate event types and context features into the softmax layer for trigger word recognition and event classification. Finally, we enumerate the combinations between two tokens in the sentence and comprehensively consider the BERT

output, NER label, predicted event types and attention results to fill the trigger word-argument role table. We explain the model details in the following subsections.

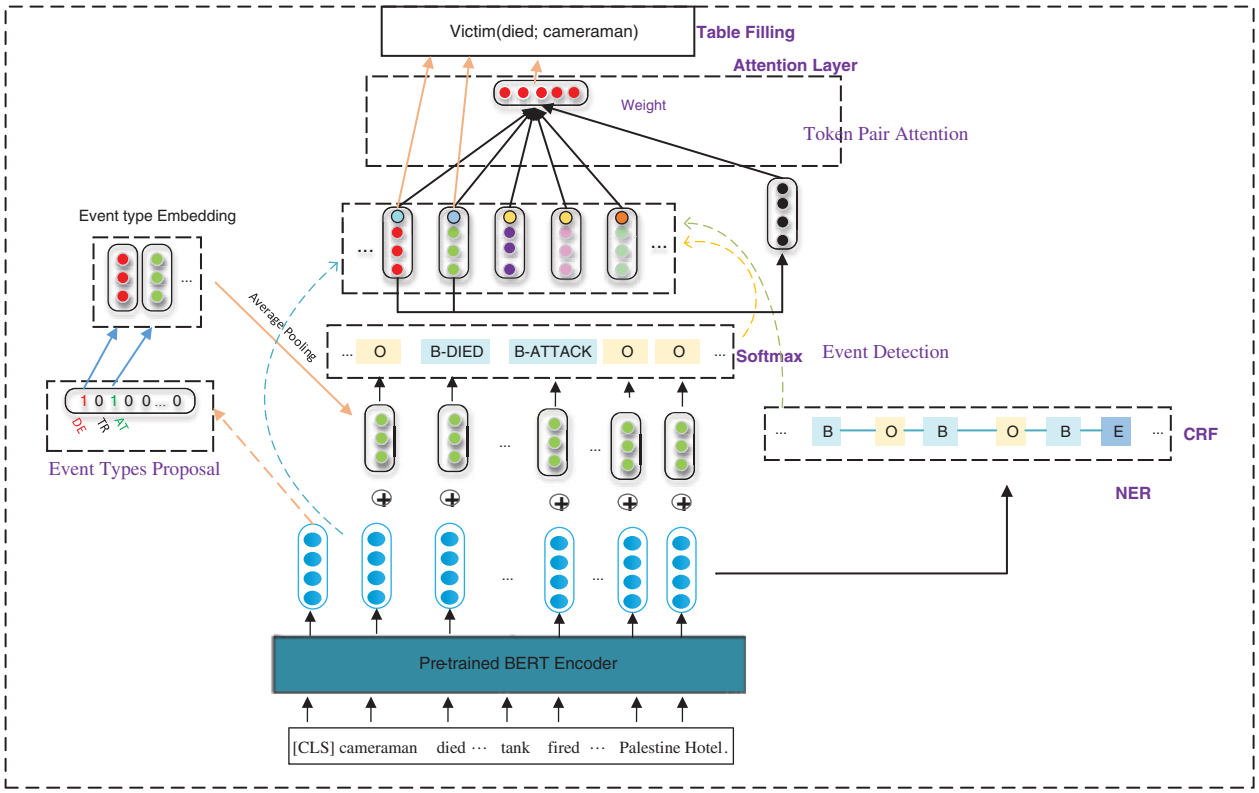

Figure 3: The overall architecture of the global-event-type guidance and attention enhancement

3.1 BERT

BERT's model architecture is a multiple layer bidirectional Transformer encoder. The encoder is as tack of identical blocks (BERT-Base stacks comprising 12 blocks on top of each other). Each block is composed of a multi-head self-attention layer and a position-wise, fully connected feed-forward layer. Assuming the output sequence of the previous layer is packed together into a matrix H, the output matrix Z of a multi-head self-attention layer is computed as

$$Z = [T_1, T_2, \ldots, T_{h-1}, T_h]\, W^O \tag{1}$$

$$T_i = softmax\left(\frac{\left(HW_i^Q\right)\left(HW_i^K\right)^T}{\sqrt{d_k}}\right)\left(HW_i^V\right) \tag{2}$$

where h is the number of attention heads, d_k is the dimension of queries and keys, and W^O, W_i^Q, W_i^K, W_i^V are the parameter matrices. Each layerin the encoder has a residual connection around it, followed by layer normalization.

For a given token, the input representation of BERT is the sum in the corresponding token and segment and position embeddings. BERT uses WordPiece embeddings as token embeddings. In addition, BERT adds a special token ([CLS]) as the first token to obtain the aggregate sequence representation for a classification task and a special token ([SEP]) to distinguish between different sentences in the same input sequence. In particular, as the input sequence extracted by the joint event is only one sentence, the special token ([SEP]) is not useful in the current task.

Given an input token sequence $X = (x_0, x_1, \ldots, x_{n-1}, x_n)$, we denote the BERT contextual representation of each token as $Z = (z_0, z_1, \ldots, z_{k-1}, z_k)$. Moreover, given that the WordPiece tokenizer might split a token into several sub-tokens, we use the hidden state corresponding to the first sub-token of a given token as its contextual representation.

3.2 Named Entity Recognition

We formulate the NER task as a sequence-labeling problem and use the BIEO (Beginning, Inside, Ending, Outside) encoding scheme. A linear-chain CRF is employed to calculate the most probable tag for each token. Formally, we first derive the emission potential that comes from the sentence encoder output. The score of each token x_i for each entity tag is calculated as follows:

$$s_i = V_1 f\left(W_1 z_i + b^h\right) + b^s \tag{3}$$

where $f(.)$ is an element wise activation function (e.g., relu), $s_i \in R^d$, d is the number of encoding scheme tags, $W_1 \in R^{l\times 2m}$, $V_1 \in R^{l\times d}$ are the transformation matrixes. $b^h \in R^l$, $b^s \in R^d$ are bias vectors, l is the hidden size, and m is the output dimension of BERT. Given a sequence of tag predictions $Y = (y_1, y_2, \ldots, y_{n-1}, y_n)$, the linear-chain CRF score is defined as

$$S(W, Y) = \sum_{i=1}^{k} (s_{i,y_i} + a_{y_{i-1},y_i}) \tag{4}$$

where s_{i,y_i} is the score of the tag y_i for the token x_i, which is obtained by Eq. (3), and a_{y_{i-1},y_i} is the score of transition from tag y_{i-1} to tag y_i. Using Eq. (4), we can get the score of a tag sequence $\boldsymbol{y}$, which is further converted to probability by the following softmax function:

$$P(Y \mid W) = \frac{\exp(s(W, Y))}{\sum_{\tilde{Y}} \exp\left(W, \tilde{Y}\right)} \tag{5}$$

where $\tilde{Y} \in Y(w)$ denotes the set of possible tag sequences for $\boldsymbol{x}$. The loss function of the sequence labeling is:

$$\mathcal{L}^{NER} = -\sum_{W\in \mathrm{T}} \log\left(Y = Y^* \mid W\right) \tag{6}$$

where T represents the training set and Y^* is the gold standard for sequence $\boldsymbol{x}$. During training, we minimize the negative log likelihood $\mathcal{L}^{NER}$ of the gold standard. In the decoding process, the Viterbi algorithm is adopted to derive the optimal tag sequence. The tags are converted to embeddings by looking up an embedding layer. We then obtain the label-embedding sequence $e^{ner} = (e_1^{ner}, e_2^{ner}, \ldots, e_n^{ner})$, $e_i^{ner} \in R^{m'}$, where m' is the dimension of the label embeddings.

3.3 Event-Types Proposal

Event-types proposal is an auxiliary task for event extraction. The task aims to predict the possible event types in the sentence regardless of which trigger word contains them. The event-types proposal layer employs hard parameter sharing, the most common approach used in multi-task learning, to share the same sentence encoder with NER. We use the first token that BERT outputs and then use a dense layer with non-linear activation to get predicted event types in this sentence:

$$E_{tp} = f\left(W_p z_0 + b_p\right) \tag{7}$$

where z_0 is the first for the BERT output. $W_p \in R^{|tp| \times h}$ is the transformation matrix, $b_p \in R^{|tp|}$ is the bias vector, $|tp|$ is the number of predefined event types, and $f(.)$ stands for the sigmoid function, which potentially allows multiple events to exist in the same sentence. We create a criterion that measures the binary cross entropy between the target and the output. The loss function of the event-types proposal is:

$$\mathcal{L}^{ETP} = -\sum_{W \in T} \sum_{i \in \nabla} \log p_{tp}\left(E_{tpi} = E^*_{tpi} \mid W\right) \tag{8}$$

where T represents the training set, ∇ is the gold standard event types set for sequence w, and E^*_{tpi} is the i-th one. $p_{tp}\left(E_{tpi} \mid W\right)$ is calculated by applying the softmax function across event types. All event types are converted to embeddings by looking up an embedding layer $e^{tp} = \left(e^{tp}_0, \ldots, e^{tp}_{\nabla}\right)$, where the average-pooling operation is e^{tp}.

$$\overline{e^{tp}} = \frac{\sum_{i=0}^{k \leq \nabla} e^{tp}_i}{k} \tag{9}$$

The predicted event types have two uses. On the one hand, the event-types proposal is a simple auxiliary task that can cooperate with the event detection task. On the other hand, it creates complex dependencies for the event types in a sentence.

3.4 Event Detection

Assume we have extracted an entire trigger candidate that meets an O label after an I-Type label or a B-Type label. Through softmax, the event-type embeddings are concatenated with the BERT contextual representation. Then we can get every token label category.

$$y_c = softmax(W_c\left[Z : expand\left(\overline{e^{tp}}\right)\right] + b_c\right) \tag{10}$$

where $W_c \in R^{m \times d}$ is the parameter matrix, $b_c \in R^m$ is the bias vector, and *expand* is the dimensional extension function. According to the obtained label probability distribution, the event-type prediction label corresponding to each token can be obtained.

$$\hat{y} = argmax\left(y_c\right) \tag{11}$$

The loss function is the cross-entropy between the target and the output for tokens:

$$\mathcal{L}^{ED} = \sum_{i=1}^{T} \log p\left(y_i \mid x_i\right) \tag{12}$$

where T represents the training set. The tags are converted to embeddings by looking up an embedding layer. We obtain the label embeddings sequence $e^{ed} = (e_1^{ed}, \ldots, e_n^{ed})$, $e_i^{ed} \in R^{m'}$, where m' is the dimension of label embeddings.

3.5 Token Pair Attention

The filling of the vanilla table takes into account just two candidate tokens to predict the role of the argument. We use the token pair attention mechanism to capture information between trigger words and arguments. Specifically, the attention score of the token pair $< x_i; x_j >$ for the k-th token in a given sentence is calculated by the following equation:

$$a_{ij}^t = \begin{cases} \dfrac{\exp(\overline{V_{ij}} W_q V_t^e)}{\sum_i \exp(\overline{V_{ij}} W_q V_t^e)} & t \neq i \text{ or } j \\ 0 & otherwise \end{cases} \tag{13}$$

where $\overline{V_{ij}}$ is the average of V_i^e and V_j^e; W_q is the attention parameter; a_{ij}^t is equal to 0 when t is equal to i or j. The main reason for this strategy is that we consider the representations of the token pair $< x_i; x_j >$ in the table filling stage (see Eqs. (15) and (16)). Thus, we directly mask the token pair themselves when performing attention calculations. The attentive result for the token pair $< x_i; x_j >$ is computed by the following equation:

$$s_{ij} = a_{ij} V^e \tag{14}$$

where $a_{ij} = (a_{ij}^1, a_{ij}^2, \ldots, a_{ij}^k)$. In the weighted average sentence representation, s_{ij} focuses on useful contextual information for table filling.

3.6 Table Filling

For event embedding (e^{ed}), NER embedding (e^{ner}) is then concatenated with the BERT contextual representation to form a final feature representation $V = (V_1^e, \ldots, V_k^e) = \left[e^{ed} : e^{ner} : Z\right]$. Let x_i and x_j be two words, Y (x_i, x_j) be all possible role relations, and $s\left(x_i, x_j, r\right)$ be a scoring function that assesses x_i and x_j for the existing role types r. We can further get the conditional probability of role types r given x_i and x_j through the softmax function:

$$p_t\left(r \mid x_i, x_j,\right) = softmax\left(s\left(x_i, x_j, r\right)\right) \tag{15}$$

$$s\left(x_i, x_j, r\right) = W\delta\left(UV_i^c + VV_j^c + Ms_{ij}\right) \tag{16}$$

Here, $\delta(.)$ is an elementwise non-linear activation function (e.g., tanh). Moreover $\{W, U, V, M\}$ are the transformation matrixes.

Based on the probability distribution of table filling, the predictive effect of each table in table filling is defined as

$$\widetilde{p_t} = argmax\left(p_t\left(r \mid x_i, x_j\right)\right) \tag{17}$$

The loss function is the cross entropy between the target and the output for all the table cells:

$$\mathcal{L}^{TF} = -\sum_{w \in T} \sum_{i=0}^{k} \sum_{j=0}^{k} \sum_{r \in \mathcal{R}} \log p_t\left(r \mid x_i, x_j, \boldsymbol{x}\right) \tag{18}$$

where $\boldsymbol{x}$ represents a sentence in the training set T, x_k is the k-th word of $\boldsymbol{x}$, n is the sentence length, and $\mathcal{R}$ is the argument-roles set between x_i and x_j.

In our framework, there are two main tasks: event detection and table filling, and two auxiliary subtasks: NER and ETP. The loss function is the summation of these four tasks. For joint event extraction, the loss function is the summation of these four tasks: $\mathcal{L}^{ED} + \mathcal{L}^{ETP} + \mathcal{L}^{NER} + \mathcal{L}^{TF}$. The loss is calculated as the average over-shuffled minibatch, and the derivatives of each parameter can be computed via backpropagation.

4 Experiments

4.1 Experiment Settings

4.1.1 Dataset, Resources and Evaluation Metric

We evaluated our framework on the ACE 2005 dataset. The ACE2005 dataset annotates 33 event subtypes and 36 role classes, and along with the NONE class and BIO annotation schema, we divided each token into 67 categories in event detection and 37 categories in argument extraction. In order to be consistent with previous work, we used the same data split as in previous work [2,7–9]. This data split included 40 newswire articles (881 sentences) for the test set, 30 other documents (1,087 sentences) for the development set and 529 remaining documents (21,090 sentences) for the training set.

We then used precision, recall, and F1 score to evaluate the performance as done in previous work [2,8,9].

4.1.2 Hyperparameter Setting

For all the experiments below, in the ETP layer and event detection, we used 200 dimensions and 300 dimensions for the tag embeddings. We utilized a maximum length $n = 120$ of sentences in the experiments by padding shorter sentences and cutting off longer ones. The batch size in our experiments was 64, we set the dropout rate to 0.5 and the learning rate to 0.05. Adam was used to optimize the neural networks. The experiments were trained with an NVIDIA RTX 1080Ti GPU.

4.2 Baselines and Main Results

To evaluate the performance of the proposed method, we compared our model with four competitive baselines, as follows: 1)DMCNN [2], which uses dynamic multiple pooling to keep multiple event information; 2) JRNN [7], which uses a bi-directional RNN and manually designed features to jointly extract event triggers and arguments; 3) dbRNN [8], which adds dependency bridges over bi-directional LSTM for event extraction; 4) JMEE [9] via attention-based graph information aggregation for multiple event extraction.

Tab. 1 shows the results of comparing our model with the baseline methods. Our framework achieved the best F1 scores in trigger recognition and trigger classification, is the scores were 1.6% higher than the best-reported models. However, it did not achieve the best F1 score for argument role classification. In summary, these results demonstrated the effectiveness of our method to incorporated with the global event type guidance and attention enhancement.

4.3 Effect of ETP Layer for Extracting Multiple Events

To evaluate the effect of the ETP layer for alleviating the multiple event phenomenon, we divided the test data into two parts (1/1 and 1/N) following [2,9,10] and then performed evaluation

separately. Here, 1/1 means that one sentence only had one trigger or one argument playing a role in one sentence; otherwise, 1/N was used.

Table 1: Overall performance comparing to the state-of-the-art methods with gold-standard entities

Method	Trigger identification (%)			Trigger classification (%)			Argument identification (%)			Argument role (%)		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DMCNN	80.4	67.7	73.5	75.6	63.6	69.1	68.8	51.9	59.1	62.2	46.9	53.5
JRNN	68.5	75.7	71.9	66.0	73.0	69.3	61.4	64.2	62.8	54.2	56.7	55.4
dbRNN	N/A			74.1	69.8	71.9	71.3	64.5	67.7	66.2	52.8	58.7
JMEE	80.2	72.1	75.9	76.3	71.3	73.7	71.4	65.6	68.4	66.8	54.9	60.3
Ours	**81.0**	**73.1**	**76.8**	**77.1**	**73.5**	**75.3**	**72.1**	**65.9**	**68.8**	65.4	**55.4**	59.9

Tab. 2 illustrates the performance (F1 scores) of JMEE [9], JRNN [7], DMCNN [2], and our framework (with and without ETP layer) in the trigger classification subtask and argument role classification subtask. From the table we can see that our framework with the ETP layer achieved the best F1 scores, they were 1.6% higher than those of the best-reported models. However, the F1 scores decreased from 75.3% to 73.1% when without the ETP layer. The results indicate that the proposed ETP layer is effective.

Table 2: System performance on single event sentences **(1/1)** and multiple event sentences **(1/N)**

Stage	Model	1/1	1/N	All
Trigger	DMCNN	74.3	50.9	69.1
	JRNN	75.6	64.8	69.3
	JMEE	75.2	72.7	73.7
	Ours	74.7	71.2	73.1
	Ours + ERP	**75.9**	**74.4**	**75.3**
Argument	DMCNN	54.6	48.7	53.5
	JRNN	50.0	55.2	55.4
	JMEE	**59.3**	**57.6**	**60.3**
	Ours	59.1	57.2	59.9

4.4 Analysis of Attention Mechanism

As Tab. 3 shows the results were not ideal when table filling alone was used for event argument extraction. The F1 value increased by 5.2% when the attention mechanism was added to the table filling, indicating that the attention mechanism can help improve the event extraction of arguments.

We used the sentence "In Baghdad, a cameraman died when an American tank fired on the Palestine Hotel" as an example to illustrate the capture feature in our attention mechanism by attention scores for the heat map in Fig. 2. There were two events in the sentence: an **Attacked**

event triggered by *fired* and a **Die** event triggered by *died*. Additionally, the entities Baghdad, Cameraman, Tank, and Palestine Hotel played an important role in **Die** and **Attacked**.

Table 3: Experimental comparison of table filling and table filling plus attention mechanism

Method	P	R	F1
Table_Filing	55.3	54.3	54.7
Table_Filing + att	59.1	57.2	59.9

As Fig. 4 shows, the trigger words "died" and "fired" had relatively strong connections with Baghdad, Cameraman, Tank, and Palestine Hotel in the **Die** and **Attacked** event, which may potentially be because of the capture information between triggers and the arguments through the attention mechanism.

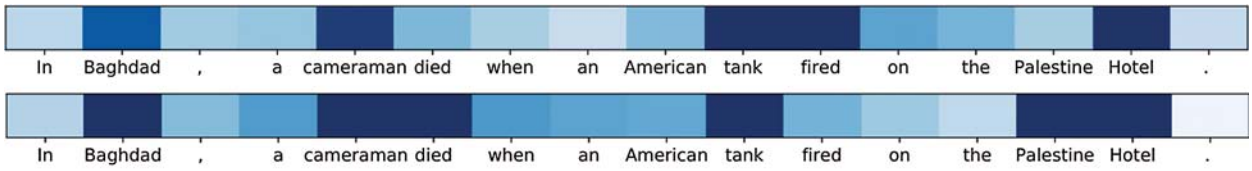

Figure 4: Illustration of token pair attention

5 Conclusion

In this work, we proposed a global event type guidance and attention enhancement to improve event detection and argument extraction. The enhancement exploits the pre-predicted event types to guide event detection, which strengthens the interdependencies of relations of multiple events in a sentence. Moreover, for argument extraction, we use the table filling method with the attention mechanism to obtain the correlation information between triggers and arguments. The experimental results on the ACE 2005 dataset indicate that our proposed model is effective, which superior to several strong baseline methods.

As the relationship between arguments among multiple events has not yet been considered, we will examine its influence on the extraction of arguments in the future.

Funding Statement: This work was supported by the Hunan Provincial Natural Science Foundation of China (Grant No.2020JJ4624, 2019JJ50655), the Scientific Research Fund of Hunan Provincial Education Department (Grant No. 19A020), and the National Social Science Fund of China (Grant No. 20&ZD047)

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] X. Ma, Y. Lu, Y. Lu, Z. Pei and J. Liu, "Biomedical event extraction using a new error detection learning approach based on neural network," *Computers, Materials & Continua*, vol. 63, no. 2, pp. 923–941, 2020.

[2] Y. Chen, L. H. Xu, K. Liu, D. J. Zeng and J. Zhao, "Event extraction via dynamic multi-pooling convolutional neural networks," in *Proc. of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th Int. Joint Conf. on Natural Language Processing of the Asian Federation of Natural Language Processing*, Beijing, China, pp. 167–176, 2015.

[3] Y. Bengio, R. Ducharme, P. Vincent and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, no. 02, pp. 1137–1155, 2003.

[4] J. Turian, L. Ratinov and Y. H. Bengio, "Word representations: A simple and general method for semi-supervised learning," in *Proc. of the 48rd Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, pp. 384–394, 2010.

[5] T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. of the 1st Int. Conf. on Learning Representations*, Scottsdale, Arizona, USA, pp. 1–12, 2013.

[6] Q. Li, H. Ji and L. Huang, "Joint event extraction via structured prediction with global features," in *Proc. of the 51rd Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria, pp. 73–82, 2013.

[7] T. H. Nguyen, K. H. Cho and R. Grishman, "Joint event extraction via recurrent neural networks," in *Proc. of the 17th Annual Conf. of the North American Chapter of the Association for Computational Linguistics*, San Diego, California, USA, pp. 300–309, 2016.

[8] L. Sha, F. Qian, B. B. Chang and Z. F. Sui, "Jointly extracting event triggers and arguments by dependency-bridge RNN and tensor-based argument interaction," in *Proc. of the 32nd AAAI Conf. on Artificial Intelligence*, New Orleans, Louisiana, USA, pp. 5916–5923, 2018.

[9] X. Liu, Z. C. Luo and H. Y. Huang, "Jointly multiple events extraction via attention-based graph information aggregation," in *Proc. of the 2018 Conf. on Empirical Methods in Natural Language Processing*, Brussels, Belgium, pp. 1247–1256, 2018.

[10] J. Bingel and A. Søgaard, "Identifying beneficial task relations for multi-task learning in deep neural networks," in *Proc. of the 15th Conf. of the European Chapter of the Association for Computational Linguistics*, Valencia, Spain, vol. 2, pp. 164–169, 2017, Short Papers.

[11] M. Miwa and Y. Sasaki, "Modeling joint entity and relation extraction with table representation," in *Proc. of the 19th Conf. on Empirical Methods in Natural Language Processing*, Doha, Qatar, pp. 1858–1869, 2014.

[12] D. J. Zeng, Y. Dai, F. Li, J. Wang and A. K. Sangaiah, "Aspect based sentiment analysis by a linguistically regularized CNN with gated mechanism," *Journal of Intelligent & Fuzzy Systems*, vol. 36, no. 5, pp. 3971–3980, 2019.

[13] D. J. Zeng, Y. Xiao, J. Wang, Y. Dai and A. K. Sangaiah, "Distant supervised relation extraction with cost-sensitive loss," *Computers, Materials & Continua*, vol. 60, no. 3, pp. 1251–1261, 2019.

[14] R. Grishman, D. Westbrook and A. Meyers, "NYU's English ACE 2005 system description," in *ACE 2005 Evaluation Workshop*, Washington, D.C, USA, pp. 1927–1938, 2005.

[15] D. Ahn, "The stages of event extraction," in *Proc. of the Workshop on Annotating and Reasoning about Time and Events for Computational Linguistics*, Sydney, Australia, pp. 1–8, 2006.

[16] S. Riedel, H. W. Chun, T. Takagi and J. Tsujii, "A markov logic approach to bio-molecular event extraction," in *BioNLP 2009 Workshop*, Boulder, Colorado, USA, pp. 41–49, 2009.

[17] H. Poon and L. V. anderwende, "Joint inference for knowledge extraction from biomedical literature," in *Proc. of the 11th Annual Conf. of the North American Chapter of the Association for Computational Linguistics*, Los Angeles, California, USA, pp. 813–821, 2010.

[18] D. Venugopal, C. Chen, V. Gogate and V. Ng, "Relieving the computational bottleneck: Joint inference for event extraction with high dimensional features," in *EMNLP*, Doha, Qatar, pp. 831–843, 2014.

[19] Q. Li, H. Ji, Yu Hong and S. J. Li, "Constructing information networks using one single model," in *Proc. of the 19th Conf. on Empirical Methods in Natural Language Processing*, Doha, Qatar, pp. 1846–1851, 2014.

[20] S. Riedel and A. McCallum, "Robust biomedical event extraction with dual decomposition and minimal domain adaptation," in *BioNLP Shared Task 2011 Workshop*, Portland, Oregon, USA, pp. 46–50, 2011.

[21] L. Y. Xiang, S. H. Yang, Y. H. Liu, Q. Li, C. Z. Zhu *et al.,* "Novel linguistic steganography based on character-level text generation," *Mathematics*, vol. 8, no. 9, pp. 1558, 2020.

[22] Z. Yang, S. Zhang, Y. Hu, Z. Hu and Y. Huang, "VAE-Stega: Linguistic steganography based on variational auto-encoder," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 880–895, 2021.