Event Extraction by Associating Event Types and Argument Roles

Qian Li, Shu Guo, Jia Wu, Senior Member, IEEE, Jianxin Li, Jiawei Sheng, Hao Peng, and Lihong Wang

Abstract—Event extraction (EE), which acquires structural event knowledge from texts, can be divided into two sub-tasks: event type classification and element extraction (namely identifying triggers and arguments under different role patterns). As different event types always own distinct extraction schemas (i.e., role patterns), previous work on EE usually follows an isolated learning paradigm, performing element extraction independently for different event types. It ignores meaningful associations among event types and argument roles, leading to relatively poor performance for less frequent types/roles. This paper proposes a novel neural association framework for the EE task. Given a document, it first performs type classification via constructing a document-level event graph to associate sentence nodes of different types and adopting a document-aware graph attention network to learn sentence embeddings. Then, element extraction is achieved by building a new schema of argument roles, with a type-aware parameter inheritance mechanism to enhance role preference for extracted elements. As such, our model takes into account type and role associations during EE, enabling implicit information sharing among them. Experimental results show that our approach consistently outperforms most state-of-the-art EE methods in both sub-tasks, especially at least 2.51% and 1.12% improvement of the event trigger identification and argument role classification sub-tasks. Particularly, for types/roles with less training data, the performance is superior to the existing methods.

Index Terms—Event extraction, argument roles, graph attention network, new schema, parameter inheritance.

I. INTRODUCTION

IN THE era of information explosion, there is an urgent need for technology to capture the critical information of aiming

Manuscript received 19 August 2022; revised 15 April 2023; accepted 26 June 2023. Date of publication 3 July 2023; date of current version 13 November 2023. The authors of this paper were supported in part by the NSFC under Grants U20B2053 and 62106059. This work was supported in part by the ARC under Grant DP230100899. Recommended for acceptance by M. Qiu. (Corresponding author: Jianxin Li.)

Qian Li and Hao Peng are with the Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing 100083, China (e-mail: liqian@act.buaa.edu.cn; penghao@act.buaa.edu.cn).

Jianxin Li is with the Zhongguancun Lab and Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing 100083, China (e-mail: lijx@buaa.edu.cn).

Shu Guo and Lihong Wang are with the National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China (e-mail: guoshu@cert.org.cn; whl@isc.org.cn).

Jia Wu is with the School of Computing, Macquarie University, Sydney, NSW 2109, Australia (e-mail: jia.wu@mq.edu.au).

Jiawei Sheng is with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100083, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100083, China (e-mail: shengjiawei@iie.ac.cn).

Digital Object Identifier 10.1109/TBDATA.2023.3291563

events from numerous texts [1], [2], [3], [4]. Event extraction technology can help us locate the text of a specific event type and find the essential arguments of the event from the text. Therefore, many scholars have studied neural network-based event extraction technology [5], [6], [7].

Event extraction (EE) is a task that acquires semantic and structural knowledge (i.e., events) from texts. Each event is represented by a typed phrase containing a trigger word, which mainly indicates the occurrence of the event, and some other arguments to ensure semantic completeness. A typical EE task usually involves two main sub-tasks: type classification is to distinguish the specified event types of sentences, and element extraction is to extract the elements including triggers and arguments under different role patterns (i.e., schemas). As shown in Fig. 1, an example sentence has two types of events for Die and Attack. For the Die event, “died” is the Trigger, and arguments of “Baghdad”, “cameraman” and “American tank” take on the roles of Place, Victim and Instrument respectively. For the Attack event, “fired” is the Trigger, and arguments of “Baghdad”, “American tank” take on the roles of Place, and Instrument respectively, and argument of “cameraman” and “Palestine Hotel” take on the roles of Target. From the example, we can see that the corresponding elements need to be extracted with their own extraction schemas for different event types.

Previous work [5], [6], [8] on neural network-based EE usually follows an isolated learning paradigm due to the distinct extraction schemas (i.e., role patterns) associated with each event type, performing extraction independently for different event types. This may lead to low performance of event extraction models for those less frequent types that do not possess enough contextual information. We argue, however, that it will be beneficial for EE to associate closely related types collectively. For the event type classification task, different event types in various sentences can be highly related. For example, a document may have both the event type Execute and Attack, which describe the same topic despite belonging to different sentences. In real event extraction datasets such as ACE 2005 [9], 123,6 sentences belong to the type Attack, but only 19 sentences belong to the type Execute. Existing methods [10], [11] classify the event types separately for each sentence in a document, failing to handle these connections among the types explicitly. Connecting these two sentences to learn associations among the two types is helpful and can provide more complementary contextual information for the less frequent type Execute.
In the context of event types, we design a document-level event graph [19] [26] to represent the most popular data set for schema-based event extraction. Always treating these semantically related arguments as different ones and extracting each element independently with its own role pattern. This also negates relevant information available in different roles. As a result, less frequent types/roles are performing poorly because the model ignores meaningful associations between event types and argument roles.

To deal with the above issues, we propose a novel neural network-based event extraction framework that learns associations among event types and relevant information in argument roles, referred to as associated EE (AEE). A given document, we need to perform EE for all sentences. First, the type classification task is conducted in a graph setting. We design a document-level event graph with both sentences and words represented as nodes. Four kinds of relationships among sentences and words are represented as edges. Then, we employ a document-aware graph attention network to learn sentence embeddings and perform classification tasks on these learned embeddings. The constructed graph has the edge between different events which appear in the same document. Thus the document-aware graph attention network (DGAT) can learn the context sharing among sentences across different types by aggregation of neighbor information. It captures the relationships among sentences and words in a document to improve the event classification task and utilize association among event types. In this way, AEE couples the type classification tasks of multiple sentences jointly in a unified event graph and enables implicit context sharing among sentences across different types. Then for the element extraction task, we design a new schema on ACE 2005 for constructing associations among argument roles. The schema couples highly related argument roles into the same one and afterward preserves all roles in one universal pattern for all event types. As such, different roles can also share knowledge jointly in a pattern. A type-aware parameter inheritance method is further introduced to draw on the type knowledge from event classification to help the extraction model focus on learning type-preferred roles in the new argument role schema. Event types are needed as the input to the argument extraction model, leading to difficulties in introducing the type-aware parameter inheritance mechanism in the original schema. By modeling correlations among types and roles together, our approach AEE may further enrich contextual information across different types/roles and alleviate the problem of low accuracy in EE caused by uneven data distribution. To be specific, we contribute to

- We propose a novel neural association event extraction framework that learns shared information among event types for the event classification task and argument roles for element extraction.
- As to event types, we design a document-level event graph and use a document-averse graph attention network to learn connections among sentence nodes. As to argument roles, a new argument role schema is built to utilize the same role patterns of all event types, realizing knowledge sharing among roles. A type-aware parameter inheritance mechanism is further developed to facilitate type-preferred role identification for different event types.
- Experimental results show that our approach consistently outperforms most state-of-the-art event extraction methods. Particularly, our method can get a more steady performance than the existing methods, especially on types/roles with fewer data.

II. RELATED WORK

The purpose of event extraction is to capture the event types that we are interested in from many texts and show the essential arguments of events in a structured form [14], which is different from unsupervised methods [15]. The event extraction task is divided into open-domain-based event extraction [5], [16], [17] and schema-based event extraction [8], [10], [18], [19] according to whether to construct an event schema for extracting triggers and arguments. Most of the existing open domain event extraction studies event extraction in social networks [20], [21], [22]. There are many kinds of events in social networks, and new event types will be derived according to the change of time. Although the method based on social network extraction is not suitable for social network events extraction. Therefore, many scholars study open-domain event extraction [17], [23]. The open domain event extraction method obtains topics by clustering or classification and then extracts event arguments from multiple texts related to the topic [16]. At present, the mainstream method is to first cluster the text and then extract keywords from each type of text as event arguments [16]. The open-domain event extraction method does not need numerous data annotation work and cannot specify a specific event category [24]. However, neural network-based event extraction in the open domain mainly extracts keywords from events, which does not always extract the core arguments of the event. Therefore, how to build a neural network-based event extraction framework should not only share the information between classes but also avoid the spread of error information. In addition, there is a standard feature extraction template. It is a difficult problem to be solved in the event extraction task.

Schema-based event extraction requires classifying the event types and extracting the specified triggers and arguments under predefined schemas or role patterns [25], [26]. ACE 2005 [9] is the most popular data set for schema-based event extraction tasks. Methods in this task can be divided into two categories. The pipeline-based event extraction methods first identify triggers and classify event types [11], [12]. Then they extract arguments according to the predicted event type and
triggers [13]. In order to overcome the error propagation caused by former sub-tasks, joint-based event extraction methods are proposed [27], [28], [29]. It identifies the triggers and arguments according to candidate triggers and entities in the first stage. In the second stage, to avoid error propagation from event type, trigger classification, and argument role classification are realized simultaneously. JIMEE [30]and DBRNN [31]have been proven influential in introducing graph information into event extraction tasks. However, all of these methods depend on the result of event classification in the argument extraction task. It may limit the shared information across event types, which is disadvantageous to neural network-based event extraction with a few labeled data.

In order to overcome the propagation of error information caused by event detection results, researchers propose a neural network-based event extraction method based on joint [27], [28], [32]. This method reduces the propagation of error information by combining trigger recognition and argument extraction tasks. However, neither pipeline-based event extraction nor joint-based event extraction can avoid the impact of event-type prediction errors on the performance of argument extraction. Moreover, these methods cannot share the information of different event types and learn each type of event independently, which is disadvantageous to the neural network-based event extraction with only a small amount of labeled data.

### III. Event Extraction Framework

We design an event extraction framework AEE for learning associations among event types and elements roles, as shown in Fig. 2. The whole framework is divided into two parts: *event type classification and element extraction*. Both event detection and element extraction rely on feature representations learned by BERT to learn sentence semantic information. For event type classification, we design a document-level graph constructing connections between sentences of different types and words for each document. A document-awared graph attention network (DGAT) is designed for learning sentence node representations to obtain context information for the current sentences. For element extraction, we design a new role schema on ACE for constructing associations between element roles. All event types use the new schema, use the same element roles of different event types, and realize knowledge sharing among different event types. Furthermore, the new argument role schema can alleviate data imbalance by the combination of multiple roles. It can reduce the transmission of error information because extracting event elements does not need to use different schemes according to different event categories. The elements in the new argument role schema are mapped to the original event schema according to the different event types. Our new argument role schema can make use of the shared information between elements and alleviate data imbalance by the combination of multiple roles. However, it may mix all roles into different types during element extraction. Thus, we further design a type-awared parameter inheritance method to associate with the features learned from event type classification. It introduces the event type classification knowledge into the element extraction task. The event type helps the model focus on learning which element roles by the type-awared parameter inheritance mechanism.

#### A. Document-Aware Event Type Classification

Event type classification classifies the event type of the current sentence. BERT is a multi-layer bidirectional Transformer [33], achieving significant performance improvement on the event-type classification task [26]. We use a BERT encoder to learn the context representation and implicitly mine the associations between events. Concretely, we fed [[CLS], s, [SEP]] to BERT to encode the input sentence s for capturing the context representation of each word, where [CLS] and [SEP] are two special tokens in BERT, and N is the length of s. Then, the word vector matrix output by BERT gets more global information through the document-level event graph construction and the document-awared graph attention networks (DGAT), as shown in Fig. 3. Finally, the event type is classified through the sentence node of each sentence.

1) **Document-Level Event Graph Construction:** We construct a document-level event graph for each document, and each word is treated as a node. It is defined as $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ represent node set and edge set, respectively. $\mathcal{V}$ owns two kinds of nodes including s sentence nodes and w word nodes, then we have $s + w = |\mathcal{V}|$. We leverage lexical knowledge to concatenate nodes and introduce a sentence node to connect all nodes in a sentence [34]. There are four kinds of connection edges between nodes, as shown in Fig. 3. The first one is the word-in-word connection. The words in a lexicon are connected one by one until they are connected to the last word. The initial weight of the word-in-word edge is length-awared pointwise mutual information (LPMI)

$$LPMI(w(i), w(j)) = \log \frac{L}{N_{w(i).w(j)} M},$$

where $N_{w(i).w(j)}$, $N_{w(i).w(j)}$ are the number of sliding windows containing word $w(i)$, $w(j)$, respectively. $N_{w(s,j)}$ are the number of sliding windows containing both $w(i)$ and $w(j)$, and $i, j \in [1, N]$. $M$ is the total number of sliding windows in the corpus. $L$ is the length of each lexicon.

The second connection is to create an edge between the lexicon. The edge weights between lexicons are measured by lexicon co-occurrence frequency

$$LCOF(l(i), l(j)) = \log \frac{L_{l(i).l(j)} N_{l(i).l(j)}}{N_{l(i)} N_{l(j)}},$$

where $N_{l(i).l(j)}$, $N_{l(i).l(j)}$ are the frequency of $l(i)$, $l(j)$ and $N_{l(i).l(j)}$ are the frequency of lexicon co-occurrence $l(i)$ and $l(j)$. $L_{l(i), l(j)}$ are the length of $l(i)$ and $l(j)$. The concrete connection mode is that the first word of the former lexicon is connected with the first word of the latter lexicon. Each edge represents the potential characteristics of the word that may exist.

We also introduce a sentence node regarded as the representation of the sentence. It is connected to all the nodes, enabling

\[^{1}\text{We utilize the gray node to represent the sentence node, and each of the other colors corresponds to a sentence in a document.}\]
Fig. 2. The overview of our proposed framework, AEE.

Fig. 3. The architecture of the DGAT. The black side is the connection of the relay node and all words, the orange side is the connection of words, the blue side is the connection of words, and the green side is the connection of words with high co-occurrence probability.

the sentence node to learn information about each sentence. The edge weights of type2token are measured by pointwise mutual information (PMI)

$$\text{PMI}(w_i, w_j) = \log \frac{N_{w_i,w_j}N_s}{N_{w_i}N_{w_j}},$$  \hspace{1cm} (3)

where $N_{w_i}, N_{w_j}, N_{w_i,w_j}$ are the number of sliding windows containing word $w_i, w_j$ and both $w_i, w_j$, and $i, j \in [1, N]$. $N_s$ is the total number of words in sentence $s$. It gathers information on all edges and nodes and eliminates boundary ambiguity between words.

The last connection is connecting sentence nodes one by one to capture the relationship among event types across sentences. The edge weights between sentence $s(i), s(j)$ are measured through sentence similarity initialized by BERT

$$\text{PS}(s(i), s(j)) = \text{sigmoid}(\text{BERT}(s(i); s(j))).$$  \hspace{1cm} (4)

According to the above definition, we initialize the document-level event graph $G$ and establish self-loop connections for each graph node.

2) Document-Aware Graph Attention Networks: This section successively introduces the multiple types of relation, and attention components to constitute the overall document-aware graph attention networks (DGAT), which is to obtain context information for the current sentences for learning sentence node representations.

To obtain entity representations containing consistent neighbor information, we first initiate node representations as follows:

$$H^{(0)}_w = \sigma \left( H_w W_w + \sum_{u \in N(w)} E_{(w,u)} H^{(0)}_u W_u / N \right),$$  \hspace{1cm} (5)

where $W_w, W_u \in \mathbb{R}^{d \times d}$ are learnable transformation matrices. $E_{(w,u)}$ is the weight of edge between node $w$ and $u$. $N$ is the length of the sentence $s$.

Considering multiple relations (e.g., type2token and type2type) in our document-level event graph, the node embedding of this graph includes the document-level knowledge. Thus, for document-knowledge aware word learning, we propose to learn the corresponding document-level knowledge vectors with the word representation

$$H^{(0)}_w, A^l = \text{DGAT}([H^{(l-1)}_w || A^{l-1}]),$$  \hspace{1cm} (6)
where $A \in \mathbb{R}^{N \times d_w}$ is a learnable vector for the word representation in the document-level knowledge learning mechanism ($d_w$ is the dimension of the word representation).

Our document-level event graph updates the representation for each word by aggregating the information from its neighbors through the adjacency matrix. It helps to learn more context information for each word. To equip the word representations with fault tolerance ability for the document-level event graph, we update the word representation with the neighborhood random dropouts graph, utilizing the neighbors and edge information for each word. The word representations in the $l$th layer are

$$H^{l}(w) = \mathbf{W} \left[ H^{l-1}(w) \| \frac{1}{D_w} \sum_{u \in N(w)} [e^{l-1}_u \| E^{l}(w,u)] \right],$$

where $D_w$ is the degree of word $w$.

The document-aware graph attention network learns the structural representation of each node in the document. The graph message passing by document-aware graph attention network is used to obtain the optimal decision. After the document-aware graph attention network, the representation of the sentence node of $s$ is $h$. The probability of a label sequence $y = c_1, c_2, \ldots, c_k$ can be defined as follows:

$$p(y \mid s) = \frac{\exp \left( \sum_{i=1}^{k} \phi(c_{i-1}, c_i, h) \right)}{\sum_{y' \in L(s)} \exp \left( \sum_{i=1}^{k} \phi(c_{i-1}', c_i', h) \right)},$$

where $L(s)$ is the set of all arbitrary label sequences. $\phi(c_{i-1}, c_i, h) = \mathbf{W}(c_{i-1}, c_i) c_i^T + \mathbf{b}(c_{i-1}, c_i)$, $\mathbf{W}(c_{i-1}, c_i) \in \mathbb{R}^{k \times n}$ and $\mathbf{b}(c_{i-1}, c_i) \in \mathbb{R}^{k \times n}$ are the weight and bias parameters specific to the labels $c_{i-1}$ and $c_i$. $k$ is the number of event types, $n$ is the input length of text. We minimize the loss through the cross-entropy loss function

$$L_c = - \sum_i p(y_i \mid s) \log p(y_i \mid s) + (1 - p(y_i \mid s)) \log (1 - p(y_i \mid s)),$$

where $p(y_i \mid s)$ and $p(y_i \mid s)$ are the $i$th real label and the predicted label.

**B. Type-Aware Element Extraction**

For a given event type, element extraction is to extract the trigger and elements related to the event type and classify the roles these elements play. An element may play multiple roles, and a word can belong to different elements. We add multiple sets of binary classifiers to BERT, each set of classifiers serving a role to determine the range of all elements belonging to it. The probability that the word $w$ is predicted to be the start position of the role $r$ is

$$P^s_r(w) = \text{softmax} \left( W^s_r \cdot B(w) \right).$$

The probability of being end position is

$$P^e_r(w) = \text{softmax} \left( W^e_r \cdot B(w) \right),$$

where $W^s_r \in \mathbb{R}^{l \times n}$ and $W^e_r \in \mathbb{R}^{l \times n}$ are learnable parameters to detect the start/end position of the element role $r$, $l$ is the number of element roles. $n$ is the length of input text on the element extraction model, which is the same as the event type classification model to enable the element extraction model to use the type-aware parameter inheritance module in Section III-B2. $B(w)$ is the representation after BERT and Bi-LSTM.

1) **New Schema Construction:** To reduce the impact of error propagation among event types, event triggers, and arguments, we consider the generalization of event argument roles. On ACE 2005, there are some different event argument roles with similar properties in different event types, such as event argument roles Time-At-Beginning and Time-Before. We construct a new argument role schema to cover the event arguments under all event types according to the criteria: Element roles that have the same meaning and similar semantic roles are considered. The original element roles $a$ and $b$ are first represented as $B(a)$ and $B(b)$ by the type-aware element extraction encoder. Then, the criteria of whether merge them as a new role are measured by cosine distance as follow.

$$\begin{align*}
\text{similarity}(B(a), B(b)) &= \frac{B(a) \cdot B(b)}{|B(a)||B(b)|} \geq \beta, \\
T(B(a)) &\neq T(B(b)),
\end{align*}$$

where $T(B(a))$ and $T(B(b))$ are the event type of element role $a$ and $b$. For any two element roles, their cosine similarity is greater than or equal to a threshold value $\beta$ and they belong to different event types, thus they are combined into a new element role $c$. The criteria for grouping highly related argument roles into the same one depend on the semantic similarity between element roles. For example, in ACE 2005, there are two event argument roles, “Time-At-Beginning” and ”Time-Before”, which have similar properties and can be grouped into one argument role called “Time-Begin”. We merge semantically similar roles across different event types by using cosine distance as a similarity criterion, which enables information sharing among different event types.

It should be emphasized that in the ACE 2005 dataset, a one-to-one correspondence between original argument roles and their associated event types and new argument roles can be identified. This implies that the original argument roles can be uniquely determined based on the event types and new argument roles. In the model prediction stage, the updated argument roles can be uniquely mapped back to their original counterparts. For example, consider the conversion process where the original schema’s argument role was labeled as Time-Before, and in our new schema, it was labeled as Time-Begin. During the element role revert block, if the model identifies the argument role as Time-Begin, we can recover the original argument role as Time-Before based on the predicted event type of BusinessMergeOrg.

The schema we designed has 14 event argument roles, as shown in Table I. We demonstrate the corresponding relationship between the original schema with 35 event argument roles and our new event argument role schema.

Different from the traditional schema-based event extraction method, we construct a new argument role schema for all event types. First, the original argument roles are converted to our new argument role schema. We then use our event extraction model to predict the event types and arguments. Finally, we revert the
original event argument role according to the new argument role schema designed. To be specific, the new argument role schema-based extraction method compared with existing methods adds the following:

a) **Element Roles Convert Block:** We convert the element roles in the original schema with the new element roles according to the new argument role schema. The corresponding content and position of the element roles remain unchanged.

b) **Element Roles Revert Block:** After the event extraction is fully complete, we revert the predicted event element roles to the original event element roles according to the predicted event type.

2) **Type-Aware Parameters Inheritance:** In order to introduce event type classification knowledge for event element extraction, we use type-aware parameter inheritance to initialize element extraction model parameters. There are two sub-tasks in the event extraction model, namely event type classification and event element extraction. The BERT encoding layer of the event type classification model encodes the input sentences. The parameters are denoted as $\Theta^{(1)}$, and the vector of each token in the sentence is denoted as $BERT(t)$ after the training stage. When switching to the element extraction model, the token of the same sentence is the input without event type. It is the main difference from existing schema-based methods to avoid error transmission of the event type classification model.

After passing the BERT coding layer, the parameters $\Theta^{(1)}$ are fixed and denoted by random initialization as the parameters $\Theta^{(2)}$, the Bi-LSTM layer of the event element extraction model receives input from its own BERT layer and the BERT layer of the event type classification model through horizontal connection. The features are transferred between the two BERT layers through connection. The specific formula is as follows:

$$h_i^{(2)} = f \left( W_i^{(2)} h_i^{(2)} + U_i^{(1)} h_{i-1}^{(1)} \right),$$

(13)

where $W_i^{(2)} \in \mathbb{R}^{d \times n}$ is the weight matrix of the $i$th layer BERT of the element extraction model, where $d$ is the dimension of each token, $n$ is the input length on the event element extraction model. $U$ is the horizontal connection from the event type classification model $i$ – 1th layer BERT to the $i$th layer BERT of the element extraction model, and $h_i$ is the input of the $i$th layer BERT of the element extraction model.

The function $f$ is $f(x) = \max(0, x)$ to make all the input of the BERT middle layer non-negative. Through the type-aware parameter inheritance mechanism, word features trained by BERT of the event type classification model are transferred to the event element extraction model. It makes the latter model inherits prior knowledge of the former model without damaging the original task sequence.

3) **Joint Loss:** For the event extraction sub-task, we design the joint loss function to be the weighted sum of three losses, including the starting position prediction loss, the ending position prediction loss, and the event type classification loss $L_c$. The loss function of starting and ending position detection is denoted as $L_s$. $L_c$: $CE$ is the cross entropy, $R$ is the element role set, and $S$ is the input sentence, and the formula of loss can be obtained as

$$L_s = \frac{1}{|R| \times |S|} \sum_{r \in R} CE(Pr_r, yr_r),$$

(14)

$$L_c = \frac{1}{|R| \times |S|} \sum_{r \in R} CE(Pr_r, yr_r).$$

(15)

The overall loss function for optimizing the element extraction model is

$$L = \lambda_1 L_s + \lambda_2 L_c + \lambda_3 L_e,$$

(16)

where $\lambda_1, \lambda_2, \lambda_3$ are hyper-parameters to balance the event classification loss and element role extraction loss.

IV. DATASETS AND SETTINGS

The proposed extraction method, along with the compared approaches, is tested on Automatic Content Extraction (ACE) 2005 [9]. It contains 599 documents annotated with eight event types, 33 event subtypes, and 35 argument roles. The ACE dataset is divided into training, validation, and test sets at an 8:1:1 ratio. We implement our model based on BERT [40], which has 12 layers, 768-dimensional hidden embeddings, and 12 attention heads. The initial learning rate is tuned in $2e^{-5}$ for BERT parameters and $5e^{-4}$ for other parameters. The maximum sequence length is 512, the learning rate is $3e^{-5}$ with an Adam optimizer, and the maximum gradient norm for gradient clipping is set to 1.0. The model is trained for 15 epochs. Following existing event extraction works [26], [38], we thus set the batch size to 8. The max training epoch is set to 20. The optimal hyperparameters are tuned on the validation set by grid search, and we tried each hyperparameter five times. We evaluate the performance of our model and comparison models for event classification (EC), trigger identification (TI), argument identification (AI), and argument role classification (ARC) subtasks. The evaluation metrics include precision (P), recall (R), and F1.

Comparisons: We compare our extraction method with nine event extraction methods: DDBNN [31] leverages dependency bridges by tree structure to carry syntactically related information when modeling each word. JMEE [30] introduces attention-based GCN to model graph information based on syntactic structure. Joint3EE [6] is a multi-task model that performs entity recognition, trigger detection, and argument role assignment.
TABLE II
The P, R, and F1 Scores on Event Type Classification and Event Trigger Identification Sub-Tasks Were Performed on the ACE 2005 Test Set

| Model            | Event Trigger Classification | Event Trigger Identification |
|------------------|------------------------------|------------------------------|
|                  | Precision | Recall | F1   | Precision | Recall | F1   |
| DBRNN [31]       | 74.10%    | 69.80% | 71.90% | -          | -      | -    |
| JMEI [30]        | 76.30%    | 71.30% | 73.70% | -          | -      | -    |
| Joint3EE [6]     | 68.00%    | 71.80% | 69.80% | -          | -      | -    |
| GAIL-ELMO [35]   | 74.80%    | 69.40% | 72.00% | 80.20%     | 72.10% | 75.90% |
| PLMEE [26]       | 81.00%    | 80.40% | 80.70% | 84.80%     | 83.70% | 84.20% |
| Chen et al. [36] | 66.70%    | 74.70% | 70.50% | 68.90%     | 77.30% | 72.90% |
| Du et al. [37]   | 71.12%    | 73.70% | 72.39% | 74.29%     | 77.42% | 75.82% |
| MQAEE [38]       | -         | -      | -     | -          | -      | -    |
| Text2Event [39]  | 69.60%    | 74.40% | 71.90% | -          | -      | -    |
| AEE              | 81.02%    | 80.28% | 81.27% | 86.90%     | 85.68% | 86.71% |

Associated event extraction (AEE) means associating event types and argument roles. The best results are highlighted in bold, “−” means results are unavailable, and the underlined values are the second best result.

TABLE III
The P, R, and F1 Scores on Event Argument Identification and Argument Role Classification Sub-Tasks Performed on the ACE 2005 Test Set

| Model            | Event Argument Identification | Argument Role Classification |
|------------------|------------------------------|------------------------------|
|                  | Precision | Recall | F1   | Precision | Recall | F1   |
| DBRNN [31]       | 71.30%    | 64.50% | 67.70% | 66.20%     | 52.80% | 58.70% |
| JMEI [30]        | 71.40%    | 65.60% | 68.40% | 66.80%     | 54.90% | 60.30% |
| Joint3EE [6]     | 59.90%    | 59.80% | 69.90% | 52.10%     | 52.10% | 52.10% |
| GAIL-ELMO [35]   | 63.30%    | 48.70% | 55.10% | 61.60%     | 45.70% | 52.40% |
| PLMEE [26]       | 71.40%    | 60.10% | 65.30% | 62.30%     | 54.20% | 58.00% |
| Chen et al. [36] | 44.90%    | 41.20% | 43.00% | 44.30%     | 40.70% | 42.40% |
| Du et al. [37]   | 58.90%    | 52.08% | 55.29% | 56.77%     | 50.24% | 53.31% |
| MQAEE [38]       | -         | -      | -     | -          | -      | -    |
| Text2Event [39]  | -         | -      | -     | 52.50%     | 55.20% | 53.80% |
| AEE              | 74.87%    | 65.24% | 70.35% | 67.82%     | 55.66% | 61.42% |

The best results are highlighted in bold, “−” means results are unavailable, and the underlined values are the second best result.

by shared Bi-GRU hidden representations. GAIL-ELMO [35] is an ELMo-based model that utilizes a generative adversarial network to focus on harder-to-detect events. PLMEE [26] is a BERT-based event extraction method using a pipeline manner, completing trigger identification, event classification, and argument extraction sub-tasks successively. Du et al. [37] design a question-answering method, which is expeditiously implementing data enhancement by constructing multiple questions for a single argument. Chen et. al. [36] use bleached statements to give a model access to the information contained in annotation manuals. MQAEE [38] is a multi-turn question-answering method utilizing argument relationships in the same event type by introducing a history answer. Text2Event [39] utilizes a sequence-to-structure generation paradigm to extract events from the sentence.

V. EXPERIMENTS AND RESULTS

A. Experimental Results on ACE 2005

Table II reports the performance of our model on three evaluation metrics of trigger classification and event trigger identification sub-tasks. To make our model and baseline model adopt the same evaluation index in trigger classification, we use the predicted triggers in the argument extraction model together with the text as the input of the event classification model to predict the event type corresponding to the current triggers. Compared to Du et al. [37], MQAEE [38] and PLMEE [26], our model boosts the F1-score by 8.88%, 9.57%, and 0.57% on trigger classification, respectively. It shows that our method is significantly superior to the MRC methods, which ignores the association between the event types. PLMEE [26] first identifies event triggers and then judges event types according to the triggers, which will cause error information transmission. Du et al. [37] and MQAEE [38] directly judge event type according to the text, leading to the lack of event trigger knowledge, resulting in the decline of event type classification performance. However, in our model, we establish the association relation among event types, which can make up for the information missing caused by the lack of event trigger information.

Table III shows the performance on event argument identification and argument role classification sub-tasks. Compared with the Joint3EE [6], the improvement of our model is 12.57% and 9.32% F1-score on the two sub-tasks. Our method can use the knowledge of event classification to extract event arguments, but it does not directly define the extracted argument role according to the result of event classification. It shows that our method can improve the performance of argument extraction while avoiding error information transmission. Compared to MQAEE [38], our
model improves 15.15% F1-score on AI and 8.02% on ARC. It shows that our method is significantly superior to the multi-turn MRC methods, which only utilize relationships among arguments in a type. Our model also consistently outperforms PLMEE [26], the best-performing baseline model not involving external knowledge. It boosts the F1-score by 5.05% and 3.42% on AI and ARC, respectively. The results show the importance of exploiting the relation among event types and element roles.

As shown in Tables II and III, our model achieves state-of-the-art performance on all four sub-tasks evaluated by P and F1. It indicates that our approach delivers the best overall results by utilizing argument relations cross-event type and knowledge inheritance from the event classification sub-task. Our approach delivers higher precision and F1 than other approaches and tends to have higher R than P, and suffers from low P than prior work. Our approach gives a lower R than the best-performing baseline model on the AI sub-task, but the resulting R is less than the best one, 0.36%.

B. Impact on Different Event Types

As shown in Fig. 4, we perform all event types on the event classification sub-task. The red line is the variation of sample numbers across different event types, and the number has a long-tail distribution. On all event types, our model can get at least 70% on precision score except for type Ownership and Fine. On the contrary, on PLMEE, there are seven event types where precisions are less than 50%. Our model can obtain a more steady performance cross-event type compared to PLMEE. It can improve the performance of event types with only a few samples. It demonstrates that our event classification model achieves cross-type knowledge sharing by introducing the DGAT.

C. Impact on Different Argument Roles

As shown in Fig. 5, we perform the 35 argument roles on our model and our model of removing the Parameters Inheritance (PI) module and Universal Schema (US) module. When the PI and US modules are not introduced, our model performance degrades as the sample number of argument roles decreases. It boosts performance on argument roles with few samples when we add the PI and US. Our model can get at least 60% on P. It can obtain a steady performance cross-type and improve the performance of event argument extraction in some event types with few samples. It achieves cross-type knowledge sharing by constructing a new argument role schema with the type-aware parameter inheritance mechanism.
TABLE IV
Ablation Study on Global Constraints on F1-Score

| Tasks | Event Type Classification | Event Trigger Identification |
|-------|---------------------------|-----------------------------|
| | Precision | Recall | F1 | Precision | Recall | F1 |
| AEE | 83.34% | 81.96% | 82.37% | 86.90% | 85.68% | 86.71% |
| -Document-aware Graph Attention Networks (DGAT) | 80.34% | 79.50% | 80.37% | 86.21% | 84.19% | 86.33% |
| -Parameters Inheritance (PI) | - | - | - | 86.01% | 83.05% | 85.27% |
| -PI-Universal Schema (PI-US) | - | - | - | 85.21% | 82.43% | 84.10% |
| -Joint Loss (JL) | - | - | - | 86.93% | 84.62% | 86.56% |

The best results are highlighted in bold, "-" means results are unavailable, and the underlined values are the second best result.

TABLE V
EVALUATION FOR VARIANTS ON EVENT TYPE CLASSIFICATION

| Initial Representation | Encoder Method | F1 | Precision | Recall |
|------------------------|----------------|----|-----------|--------|
| Random | BERT | 77.38% ± 1.66% | 77.28% ± 1.98% | 77.21% ± 1.46% |
| | GCN | 76.38% ± 1.66% | 76.28% ± 1.98% | 77.21% ± 1.46% |
| | GAT | 76.25% ± 1.70% | 76.07% ± 1.47% | 77.04% ± 1.35% |
| | DGAT | 77.29% ± 1.53% | 77.40% ± 1.82% | 77.92% ± 1.48% |
| | BERT+GCN | 77.20% ± 1.05% | 77.85% ± 1.26% | 78.60% ± 1.19% |
| | BERT+GAT | 78.19% ± 1.08% | 78.23% ± 1.60% | 78.79% ± 1.04% |
| | BERT+DGAT | 79.53% ± 1.12% | 78.52% ± 1.42% | 78.93% ± 1.32% |
| Glove | BERT | 75.75% ± 1.19% | 76.76% ± 1.29% | 76.22% ± 1.98% |
| | GCN | 74.95% ± 1.15% | 74.82% ± 1.80% | 74.17% ± 1.92% |
| | GAT | 76.32% ± 1.90% | 75.36% ± 1.81% | 75.01% ± 1.89% |
| | DGAT | 77.34% ± 1.74% | 76.03% ± 1.24% | 76.38% ± 1.50% |
| | BERT+GCN | 77.76% ± 1.41% | 76.28% ± 1.29% | 77.22% ± 1.98% |
| | BERT+GAT | 78.96% ± 1.54% | 77.06% ± 1.36% | 77.15% ± 1.63% |
| | BERT+DGAT | 79.48% ± 1.26% | 78.83% ± 1.19% | 78.23% ± 1.07% |
| BiLSTM | BERT | 63.85% ± 1.52% | 60.13% ± 1.39% | 61.71% ± 1.21% |
| | GCN | 77.85% ± 1.30% | 76.13% ± 1.36% | 77.27% ± 1.86% |
| | GAT | 77.16% ± 1.51% | 77.73% ± 1.09% | 77.58% ± 1.02% |
| | DGAT | 78.04% ± 1.37% | 77.65% ± 1.63% | 77.26% ± 1.35% |
| | BERT+GCN | 78.85% ± 1.52% | 77.13% ± 1.20% | 77.71% ± 1.21% |
| | BERT+GAT | 79.11% ± 1.77% | 78.16% ± 1.39% | 78.46% ± 1.12% |
| | BERT+DGAT | 80.53% ± 1.31% | 79.05% ± 1.69% | 79.80% ± 1.89% |
| BERT | BERT | 80.37% ± 1.09% | 80.34% ± 1.74% | 79.50% ± 1.78% |
| | GCN | 79.68% ± 1.25% | 79.28% ± 1.29% | 78.12% ± 1.78% |
| | GAT | 79.96% ± 1.06% | 80.20% ± 1.41% | 78.40% ± 1.25% |
| | DGAT | 80.84% ± 1.23% | 81.05% ± 1.35% | 79.92% ± 1.83% |
| | BERT+GCN | 80.68% ± 1.25% | 80.28% ± 1.46% | 78.12% ± 1.78% |
| | BERT+GAT | 81.43% ± 1.53% | 81.55% ± 1.72% | 79.40% ± 1.06% |
| | BERT+DGAT | 82.37% ± 1.23% | 83.34% ± 1.24% | 81.96% ± 1.83% |

The bold values are the best result and the underlined values are the second best result.

D. Ablation Study

We evaluate four variants of our approach given in Table IV. We remove the Document-level Graph Attention network (DGAT) module. F1-score decreases on all four sub-tasks, which shows that the DGAT module can significantly improve the performance of EC subtasks. It positively affects learning global knowledge and EC knowledge is helpful to element extraction. We further remove the PI module, improving element extraction by using knowledge of event classification. The performance significantly descends, with TI, AI, and ARC
sub-tasks by 1.44%, 1.38%, and 1.35% on the F1-score. It may prove that inheriting the knowledge of event classification by the PI module can effectively improve element extraction. When it comes to the US module, we need to remove the PI module meanwhile. F1-score decreases by 2.61%, 3.04%, and 2.30% compared to the AEE, respectively. It may prove that our US module can learn more relations of the argument roles. Moreover, the Joint Loss (JL) module is employed for argument extraction to comprehensively consider the loss of EC and argument extraction. It contributes to the F1-score of the TI, AI, and ARC tasks at 0.15%, 0.23%, and 0.20%. The results suggest that all variants are helpful, and the PI and US modules are the most important for the element extraction, as removing any of them can result in the most drastic performance degradation.

E. Discussion on Generalized Variants

To evaluate the effectiveness and generality of components, we experiment with various variants of our model on ACE 2005, as shown in Table V. Specifically, for words’ initial representation, we employ four methods: Random, Glove, BiLSTM, and BERT. In most circumstances, using BERT initial representation acquires the best performance through learning better context information. For the encoder, we apply BERT, BERT+GCN, BERT+GAT, and BERT+DGAT to learn word embedding. We analyze the experimental results and observe that: (1) BERT+GAT encoder is better than the BERT+GCN encoder in our event classification model. This is mainly because the GAT structure pays more attention to important words for event classification, which is more suitable for short sentences to capture both context and structure information. (2) BERT+DGAT encoder is better than the BERT+GAT encoder in the event classification model. For the BERT+DGAT encoder, the document-level event graph and document-aware graph attention networks help to learn more global information. The main reason is that event detection in short texts is more dependent on document-level knowledge. (3) In our baseline methods, the BERT-based method has better performance than all LSTM-based models and GNN-based models. Furthermore, in the BERT-based models, we try to combine the graph method and the model has a better effect. It may be because the whole model captures more useful information from context and structure for word representation and event detection.

F. Impact of DGAT Layer Number

We test the performance on event classification, trigger identification, argument identification, and argument role classification sub-tasks and observe the F1-score of our model with different DGAT layers. As shown in Fig. 6, it presents empirical evidence to demonstrate the layer effect on our document-level event graph. We observe that F1-score enhances when the number of DGAT layers increases. It shows that DGAT can capture contextual information at the document level, capturing more global knowledge than at the sentence level. DGAT is conducive to event extraction on limited data, which requires more DGAT layers to maintain the best performance. However, the improvement reaches a peak when using 5 DGAT layers across all sub-tasks. A further increase in the number of DGAT layers does not give a further advance. Therefore, we finally choose to use 5 DGAT layers.

G. Case Study

Our event extraction framework makes use of the relationship between event types and elements across types. It can recognize the element more completely by learning the relationship of element roles across event types. For example, in the sentence “As the soldiers approached, the man [...],” the word “the man” can play different roles. When the event type is Die, its role is Agent, while when the event type is Attack, its role becomes Attacker. In our new argument role schema, the two-argument roles are no longer distinguished and are unified as Initiator. Furthermore, it can reduce performance fluctuation caused by uneven data distribution under different element roles. For example, the role Price has 12 samples, the smallest one. The role is changed to Currency with 211 samples by combining role Price and Money. Thus, we can recognize the role Price...
more than ever. By designing a unified template, some samples with fewer categories are integrated into one with other types to improve the extraction ability of fewer categories. However, this design does not guarantee that the performance of small samples will be the same as that of other categories. It fails to identify arguments of scattered distribution containing multiple events. For example, the sentence “Some 70 people were arrested Saturday [...]” contains four types, all having the same role Time-Within being “Saturday”, which our model cannot identify all “Saturday”的. Our future work will look into this.

VI. CONCLUSION AND FUTURE WORKS

We propose AEE to learn shared information among event types and argument roles. We propose a document-awared graph attention network to establish type information sharing by connecting type nodes in event types. We design a new schema on ACE 2005 for constructing associations among argument roles on the element extraction task. The schema couples highly related argument roles into the same one and afterward preserves all roles in one universal pattern for all event types. Furthermore, we design a type-awared parameter inheritance mechanism to consider event-type knowledge for element extraction. Our method solves the problem that the performance is irregular when compared between classes, caused by the long-tail distribution of data. Future work considers distributed training on larger datasets and enhancing scalability [41], [42] and using reinforcement learning to learn the best extraction order of elements.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful comments and suggestions.

REFERENCES

[1] W. Xiang and B. Wang, “A survey of event extraction from text,” IEEE Access, vol. 7, pp. 173111–173137, 2019.
[2] F. Liu et al., “Deep learning for community detection: Progress, challenges and opportunities,” in Proc. 29th Int. Conf. Artif. Intell., C. Bessiere, Ed., 2020, pp. 4981–4987.
[3] X. Ma, J. Wu, S. Xue, J. Yang, Q. Z. Sheng, and H. Xiong, “A comprehensive survey on graph anomaly detection with deep learning,” 2021, arXiv:2106.07178.
[4] X. Su et al., “A comprehensive survey on community detection with deep learning,” 2021, arXiv:2105.12584.
[5] R. Wang, D. Zhou, and Y. He, “Open event extraction from online text using a generative adversarial network,” in Proc. Conf. Empir. Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process., Hong Kong, China, 2019, pp. 282–291.
[6] T. M. Nguyen and T. H. Nguyen, “One for all: Neural joint modeling of entities and events,” in Proc. 33rd AAAI Conf. Artif. Intell., 31st Innov. Appl. Artif. Intell., Conf. 9th AAAI Symp. Educ. Adv. Artif. Intell., Honolulu, Hawaii, USA, 2019, pp. 6851–6858.
[7] Y. Zhou, Y. Chen, J. Zhao, Y. Wu, J. Xu, and J. Li, “What the role is vs. what plays the role: Semi-supervised event argument extraction via dual question answering,” in Proc. 55th AAAI Conf. Artif. Intell., 33rd Conf. Innov. Appl. Artif. Intell., 11th Symp. Educ. Adv. Artif. Intell., 2021, pp. 14638–14646.
[8] J. Ferguson, C. Lockard, D. S. Weld, and H. Hajishirzi, “Semi-supervised event extraction with paraphrase clusters,” in Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol., New Orleans, Louisiana, USA, 2018, pp. 359–364.
[31] 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. 32nd AAAI Conf. Artif. Intell., 30th Innov. Appl. Artif. Intell., 8th AAAI Symp. Educ. Adv. Artif. Intell., New Orleans, Louisiana, USA, 2018, pp. 5916–5923.

[32] H. Wen et al., “RESIN: A dockerized schema-guided cross-document cross-lingual cross-media information extraction and event tracking system,” in Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.: Demonstrations, 2021, pp. 133–143.

[33] A. Vaswani et al., “Attention is all you need,” in Proc. Adv. Neural Inf. Process. Syst., Long Beach, CA, USA, 2017, pp. 5998–6008.

[34] T. Gui et al., “A lexicon-based graph neural network for chinese NER,” in Proc. Conf. Empir. Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process., Hong Kong, China, 2019, pp. 1040–1050.

[35] 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, 2019.

[36] Y. Chen, T. Chen, S. Ebner, and B. V. Durme, “Reading the manual: Event extraction as definition comprehension,” 2019, arXiv:1912.01586.

[37] X. Du and C. Cardie, “Event extraction by answering (almost) natural questions,” in Proc. Conf. Empir. Methods Natural Lang. Process., 2020, pp. 671–683.

[38] F. Li et al., “Event extraction as multi-turn question answering,” in Proc. Conf. Empir. Methods Natural Lang. Process., 2020, pp. 829–838.

[39] Y. Lu et al., “Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction,” in Proc. 59th Annu. Meeting Assoc. Comput. Linguistics 11th Int. Joint Conf. Natural Lang. Process., C. Zong, F. Xia, W. Li, and R. Navigli, Eds., Association for Computational Linguistics, 2021, pp. 2795–2806.

[40] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol., Minneapolis, MN, USA, 2019, pp. 4171–4186.

[41] X. Luo et al., “HAN: A hierarchical autotuned collective communication framework,” in Proc. IEEE Int. Conf. Cluster Comput., Kobe, Japan, 2020, pp. 23–34.

[42] D. Zhong, Q. Cao, G. Bosilca, and J. I. Dongarra, “Using long vector extensions for MPI reductions,” Parallel Comput., vol. 109, 2022, Art. no. 102871.

Jia Wu (Senior Member, IEEE) received the PhD degree in computer science from the University of Technology Sydney, Ultimo, NSW, Australia. He is currently an ARC DECRA fellow with the School of Computing, Macquarie University, Sydney, Australia. His current research interests include data mining and machine learning. Since 2009, he has published more than 100 refereed journal and conference papers, including IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE Transactions on Knowledge and Data Engineering, IEEE Transactions on Neural Networks and Learning Systems, IEEE Transactions on Multimedia, ACM Transactions on Knowledge Discovery from Data, NIPS, WWW, and ACM SIGKDD.

Jianxin Li is currently a professor with the State Key Laboratory of Software Development Environment, and Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University. His current research interests include social network, machine learning, Big Data and trustworthy computing.

Jiawei Sheng is currently working toward the PhD degree with the Institute of Information Engineering, Chinese Academy of Sciences. His current research interests include information extraction, knowledge graph embedding and knowledge acquisition.

Hao Peng is currently an assistant professor with the School of Cyber Science and Technology, and Beijing Advanced Innovation Center for Big Data and Brain Computing in Beihang University. His research interests include representation learning, machine learning and graph mining.

Lihong Wang is currently a professor with the National Computer Network Emergency Response Technical Team/Coordination Center of China. His current research interests include information security, cloud computing, Big Data mining and analytics, information retrieval, and data mining.