IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 5 March 2024, accepted 14 March 2024, date of publication 21 March 2024, date of current version 1 April 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3380415

RESEARCH ARTICLE

ES4ED: An Event Detection Model Based on Event Sentence Pre-Determination

JIZHAO ZHU[1], HAONAN ZHAO[1], WENYU DUAN[1], XINLONG PAN[2], AND CHUNLONG FAN[1]
[1]School of Computer Science, Shenyang Aerospace University, Shenyang 110136, China
[2]Institute of Information Fusion, Naval Aeronautical University, Yantai 264001, China

Corresponding author: Xinlong Pan (airadar@126.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 62076249, and in part by the Natural Science Foundation of Shandong Province under Grant ZR2020MF154.

ABSTRACT As an important task in the field of information extraction, event detection is widely used in event graph construction and network public opinion monitoring. Although the existing methods (such as BGCN, MGRN-EE, etc.) have obtained well performance on event detection by utilizing various features from text, they neglect that the events in data follows a long-tailed distribution, which leads to a serious bias in the trained event detection model. By following a simple but effective way to address this issue, we propose an event detection model based on event sentence pre-determination, termed as ES4ED. The model first employs classification method to identify the sentences that contain events semantically (called event sentences), and then conducts event detection on these event sentences to solve the long-tailed distribution of events. ES4ED consists of three components: the semantic encoder, the event sentence decider and the event detector. First, the semantic encoder encodes the words semantically. Then, the event sentence decider identifies event sentences by classification. Finally, the event sentences are input to the event detector to complete the event triggers identification and classification. Experimental results on the public dataset ACE2005 show that the F1 score of the proposed model achieves 79.2% and 76.5% on trigger identification and trigger classification, respectively, which are significantly improved compared with the existing typical works.

INDEX TERMS Event extraction, event detection, pre-determination, event sentence.

I. INTRODUCTION

Events are one of the essential features of human society, and the social activities of people are often driven by events [1]. Aiming to reveal the evolution law and development logic of events, the team of professor Liu Ting from Harbin Institute of Technology proposed the concept of event evolutionary graph (EEG) [2], which is a knowledge base of event logic, describing the evolutionary laws and patterns between events. EEG has been applied to tasks including event prediction, intention understanding, public opinion monitoring, dialogue generation, and information retrieval [3], [4], [5], [6]. As a critical task in event extraction, event detection directly affects the quality of EEG construction. According to the Automatic Content Extraction (ACE) review conference definition, event detection aims to recognize event triggers from the given text and classify them into specific event types. In this case, the trigger is the word that most clearly express the occurrence of the event, usually verbs or nouns that indicate the change of state or action. Take the following sentence S as an example, an event detection system is able to identify the trigger "*pot shot*" and its corresponding event type "*Attack*".

S*: A man in this vehicle took a pot shot at the commandos.*

Traditional methods for event detection mainly rely on complex feature engineering, which means selecting suitable features from text to input into the classifier to complete event detection [7], [8], [9]. Ahn [7] used lexical features, dependency features, event type and other features to complete event classification through two machine learning algorithms. Chieu and Ng [8] used the maximum entropy model to complete event extraction by defining simple features such as

The associate editor coordinating the review of this manuscript and approving it for publication was Maria Chiara Caschera.

named entities and the first word in a sentence. Li et al. [9] introduced global features to improve performance, and used the dependency between trigger words and arguments to combine with local features for event extraction. Such methods are centered on manual features which rely heavily on domain expert knowledge, thus the generalization ability of these methods is weak. In recent years, methods based on deep learning have been widely used in event detection. With the capability of neural networks to learn features automatically, these methods have a stronger generalization capability than traditional event detection methods. Chen et al. [10] used dynamic multi-pooling convolutional neural networks to capture features from different intervals; Nguyen et al. [11] learned enriched sentence-level representations through bidirectional recurrent neural networks; Sha et al. [12] incorporated syntactic dependency information into recurrent neural networks. Previous works use deep neural networks to automatically learn the semantic, syntactic, and dependency features of text, which significantly improves the performance of event detection. However, in the real world, the occurrence of events is characterized by suddenness, occasionality and transience, therefore events under digital space are characterized by the long-tailed distribution in data. The current event detection methods neglect the long-tailed distribution of events in the training data, which leads to the serious bias injected into the event detection model during the training process. Taking the ACE2005[1] dataset commonly used in the field of event detection as an example, the English corpus contains 599 documents and 16,166 sentences, but only 3,867 sentences contain event information (called event sentences), which accounts for 23.9%.

In order to solve the issue of long-tailed distribution of events in data, we propose an event detection model based on event sentence pre-determination, called ES4ED. Our model employs classification method to complete the event sentences recognition, and then conducts event detection on the event sentences to resolve the long-tailed distribution of events in data. ES4ED is composed of semantic encoder, event sentence decider and event detector. Firstly, the semantic encoder encodes the words semantically, then determines whether the given sentences contain event information by the event sentence decider, and finally inputs the event sentences to the event detector to complete the identification and classification of the event triggers. Specifically, the semantic encoder first employs the pre-trained model bidirectional encoder representations from transformers (BERT) [13] to complete the encoding of words in the sentence. Moreover, the event sentence decider uses convolutional neural networks (CNNs) to capture the features under multi-scale convolution and obtain the semantic representation of the sentence through fusion, followed by binary classifier to complete the prediction of event sentence. Afterwards, the event sentence is fed into the bidirectional long short-term memory (Bi-LSTM) layer of event detector and a sequence of hidden states are outputted. We then integrate them with the embeddings of word and entity type by concatenation operation to further yield the enhanced representation of each token in the event sentence. Finally, the trigger identification and event type classification are accomplished by sequence annotation. We have done extensive experiments on the ACE2005 public dataset, and the experimental results show that the ES4ED achieves the F1 scores of 79.2% and 76.5% on the tasks of trigger identification and trigger classification, which are significant improvements compared with previous works.

In summary, the main contributions of our work are as follows:

1. To address the issue of long-tailed distribution of events in data, we propose a novel and effective event detection model based on a very simple idea, termed as ES4ED.

2. ES4ED first employs classification method to identify event sentences, and then conducts event detection on these event sentences to resolve the long-tailed distribution of events in data.

3. Extensive evaluation shows that our model is competitive on the ACE2005 public dataset and obtains significantly improved compared with previous works.

II. RELATED WORK

Event detection is a fundamental yet challenging task in information extraction, aiming at identifying event mentions from the text and classifying their corresponding types. With the development of deep learning and its successful application in various fields, deep learning technology has achieved significant results on event detection task and has become the mainstream in this field. Up to now, a large number of research works have explored various deep neural network models and techniques to improve the effectiveness of event detection, such as, CNNs, recurrent neural networks (RNNs), attention mechanism, and pre-training language model, etc. Chen et al. [10] proposed the dynamic multi-pool convolutional neural networks, which introduced a word representation model to capture lexical level features, and sentence-level clues were captured using CNNs. Sentence was divided into different intervals according to the positions of candidate trigger words and candidate arguments, and using dynamic convolution to aggregate key information in different intervals. The model can retain as much critical information as possible when considering multi-event sentences. Ramponi et al. [14] proposed a biomedical event extraction model for cross-domain edge detection combined with CNNs. Nguyen et al. [11] proposed the RNNs based model that uses bidirectional RNNs to learn richer sentence representations and introduces the memory matrix to store recognition information during tagging to capture the dependencies between trigger words and argument roles. Sha et al. [12] proposed bridge-dependent RNNs to establish corresponding links between the nodes of Bi-LSTM to enable syntactically dependent information can be propagated among the nodes of LSTM. Experiments show that

[1] https://catalog.ldc.upenn.edu/LDC2006T06

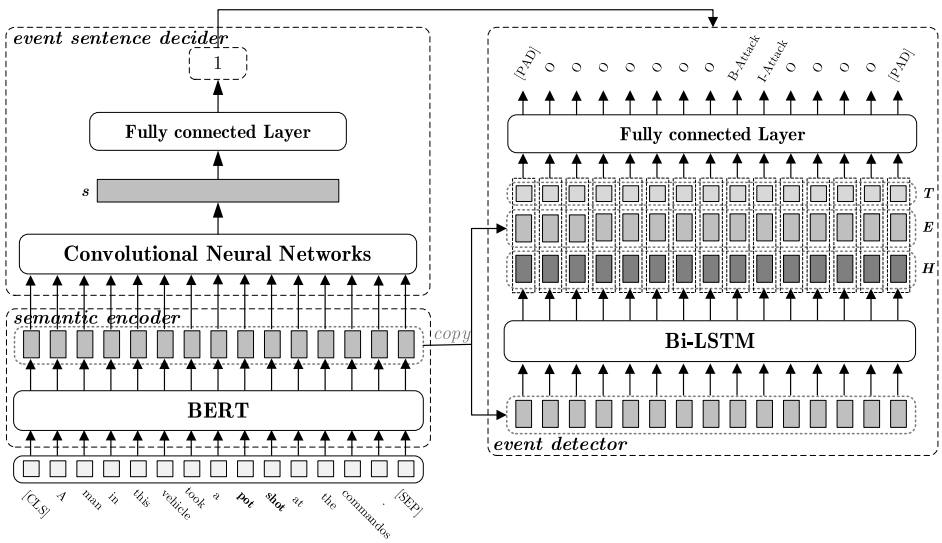

FIGURE 1. The architecture of our model ES4ED. It illustrates the processing of one instance, where the input sentence is first predicted as an event sentence, and then the event trigger "*pot shot*" and its corresponding event type "*Attack*" are extracted by sequence annotation.

combining tree and sequence structures in LSTM models possesses better performance.

To utilize the correlations between events to enhance the effectiveness of event extraction, researchers have used the graph convolutional networks (GCNs) to construct the graph of dependencies between events. Liu et al. [15] proposed a multi-event joint extraction framework, which used syntactic shortcut arcs to enhance information flow and attention-based graph convolution to model graph information and realize joint extraction of multiple event trigger words and arguments. Cui et al. [16] proposed the edge-enhanced graph convolutional networks that utilize both syntactic structure and type label information for event detection. Cheng et al. [24] proposed a trigger detection model based on BERT and GCNs, which strengthens the feature representation by introducing BERT and introduces syntactic structure to capture long-distance dependencies. With the attention mechanism proposed, researchers have introduced it to the event detection task, so that the model can focus on the key parts when processing text. Duan et al. [17] argued that with all subtasks sharing one single representation, the information needed for a specific task is ignored or underestimated, and therefore proposed the multi-grained ranking event extraction network. Each subtask constructs a multi-grained span representation from three granularities and uses two attention mechanisms to explore the association of event-event and event-entity.

In addition, Liu et al. [18] proposed a new concept of trigger salience attribution for event detection tasks, determining the extent to which events depend on triggers or depend on contexts, explicitly quantifying the patterns of events, and using a training mechanism that uses salience as evidence to enhance learning. There are also researchers have viewed event extraction tasks as question answer task [19], conditional generation task [20], and query and extraction task [21], among others. Du and Cardie [22] argued that there is too much reliance on entity information in existing event extraction methods and proposed a new event extraction paradigm. They treated the event extraction task as a question answering task with an end-to-end structure and designed question templates for trigger detection and argument extraction, respectively, without any preprocessing step for entity recognition, and experiments showed that the model performs equally well on zero-sample learning.

Although the above methods have obtained a significant improvement on the results of event detection, they neglected the fact that the events in data follows a long-tailed distribution, which leads to the serious bias of the trained event detection model. Therefore, we take the way of event sentence pre-determination, which firstly identifies the event sentences by using classification method, and then does event detection processing on the event sentences, so as to solve the issue of long-tailed distribution of events. Finally, the detection of event triggers and the classification of triggers are done by sequence annotation.

III. OUR MODEL

In this paper, we propose an event detection model ES4ED, which mainly consists of three components: the semantic encoder, the event sentence decider, and the event detector, as illustrated in Fig. 1. Firstly, the semantic encoder uses the pre-trained language module BERT to complete the semantic encoding for the words in sentence. Then, the event sentence decider captures text features with different granularity by using CNNs and fuses them to obtain a kind of semantically enhanced sentence representation, and completes the event sentence determination. Finally, the event detector takes the word embeddings of event sentence as input and obtains the hidden state sequence through the Bi-LSTM layer of event

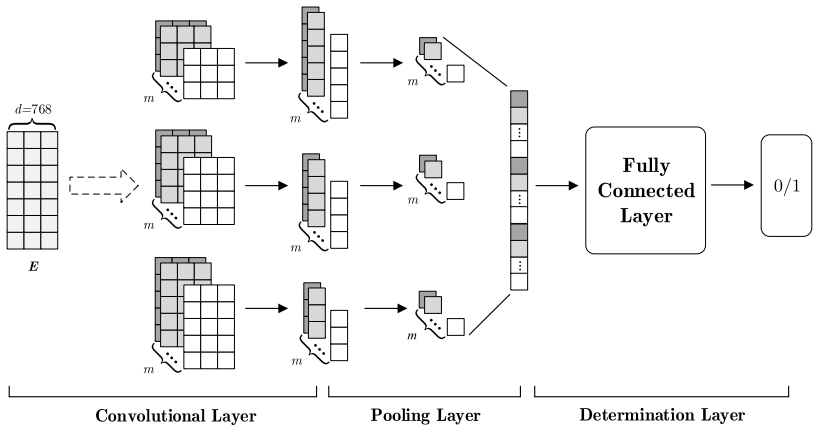

FIGURE 2. The structure of event sentence decider. d denotes the dimension of word embedding, E denotes the word embedding matrix, m denotes the number of convolution windows with different width. The output is expected to be 1 if the given sentence contains event, otherwise 0.

detector, and the merge with the corresponding word embeddings and entity type embeddings to obtain the enhanced semantic representation for each word, and then the event trigger identification and trigger classification can be done in the way of sequence annotation.

A. SEMANTIC ENCODER

We employ BERT model as the encoder for our model since it has a powerful semantic representation ability. The input sentence is denoted as $S = \{w_1, w_2, \cdots, w_n\}$, where n represents the sentence length. According to the input setting for BERT, the tokens [CLS] and [SEP] are padded to the beginning and end positions of the sentence S, and the padded sentence $S' = \{[\text{CLS}], w_1, w_2, \cdots, w_n, [\text{SEP}]\}$. The embedding for each word in a sentence can be obtained through BERT. And these embeddings are denoted as a matrix $\boldsymbol{E} = \left\{\boldsymbol{e}_{[\text{SEP}]}, \boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_n, \boldsymbol{e}_{[\text{SEP}]}\right\}$, where the dimension of word embedding is 768.

B. EVENT SENTENCE DECIDER

We adopt CNNs as the backbone network structure of the event sentence decider to process the input. The structure of event sentence decider includes convolutional layer, pooling layer and determination layer, as shown in Fig. 2.

1) CONVOLUTIONAL LAYER

To capture sentence features efficiently, convolutions with three different window width are used in the convolution process, where the heights of these windows are the same as the dimension of the word embedding, and the widths are set to 3, 4, and 5. We use m to denotes the number of convolution windows with different width in our model. The word embedding matrix $\boldsymbol{E}$ is taken as the input to the convolutional layer, and a local feature can be obtained after the convolution operation, as shown in (1):

$$c = \text{ReLU}\left(\boldsymbol{W}_1 \times \boldsymbol{E}_{i:i+h-1} + \boldsymbol{b}_1\right) \tag{1}$$

where $\boldsymbol{W}_1$ and $\boldsymbol{b}_1$ are the weight matrix and bias term; h denotes the width of convolution window; $\boldsymbol{E}_{i:i+h-1}$ denotes the h word vectors in rows i to $i+h-1$ of the word embedding matrix $\boldsymbol{E}$, where $i = [1, 2, \cdots, n-h+1]$. After the convolution layer, the complete feature representation $\boldsymbol{c} = [c_1, c_2, \cdots, c_{n-h+1}]$ is obtained.

2) POOLING LAYER

The features output from the convolutional layer are downscaled using the maximum pooling operation to capture the most important one-dimensional feature $\hat{c} = \max(\boldsymbol{c})$. The features obtained under all convolutions of the same window width are respectively subjected to maximum pooling operation and combined to get the feature vector $\hat{c}^h = \left[\hat{c}_1, \hat{c}_2, \cdots, \hat{c}_m\right]$, where $h = 3, 4, 5$. Finally, the feature vectors $\hat{c}^3, \hat{c}^4, \hat{c}^5$ are spliced to obtain the semantic encoding representation of the sentence, denoted as $\boldsymbol{s} = \left[\hat{c}^3, \hat{c}^4, \hat{c}^5\right]$.

3) DETERMINATION LAYER

The final sentence encoding $\boldsymbol{s}$ is input to the fully connected layer, and the probability value p that $\boldsymbol{s}$ is an event sentence is obtained by the sigmoid activation function, as shown in (2), where $\boldsymbol{W}_2$ and b_2 are the weight matrix and bias term.

$$p = \text{sigmoid}\left(\boldsymbol{W}_2 \times \boldsymbol{s} + b_2\right) \tag{2}$$

To train the event sentence decider, the cross entropy function is used to define the loss in the event sentence decision stage, as shown in (3):

$$L_s = -\sum_{i=1}^{N}\left[y_i \log p_i + (1-y_i)\log(1-p_i)\right] \tag{3}$$

where N denotes the training sample size; y_i is the true label of the i-th sample, which takes the value of 1 to indicate that it is an event sentence and 0 to indicate that it is not; p_i denotes the predicted probability that the i-th sample is an event sentence.

C. EVENT DETECTOR

The role of the event detector is to identify the word or phrase that triggered the event from the event sentence and to complete the prediction of the type of event triggered. To make full use of the contextual semantic information of the text, we use the Bi-LSTM layer to further process the output of the semantic encoder and obtain a set of hidden state sequences. Through the fusion of hidden state vector, word embedding and entity type feature, an information-enhanced unified representation is obtained to effectively improve the performance of event detection.

In this paper, we regard the event detection task as a sequence annotation task and use the BIO annotation method to predict the position and event type of triggers. The label "B- $*$" indicates the start position of the trigger word of the event type $*$, the label "I- $*$" indicates the middle or end position of the trigger word of the event type $*$, and the label "O" indicates a non-event trigger word.

1) BI-LSTM LAYER

For extracting the contextual semantic information of the sentence, Bi-LSTM is chosen to be used, which is able to capture the bi-directional semantic dependencies more deeply to get the hidden state sequences through the combination of forward LSTM state and reverse LSTM state. At the current time t, the forward LSTM state is denoted as $\vec{\boldsymbol{h}}_t$, the reverse LSTM state is denoted as $\overleftarrow{\boldsymbol{h}}_t$, and the final output is denoted as $\boldsymbol{h}_t = \left[\vec{\boldsymbol{h}}_t, \overleftarrow{\boldsymbol{h}}_t\right]$. The input to the Bi-LSTM layer is the word embedding matrix E, and the output is the sequence of hidden states $\boldsymbol{H} = \left\{\boldsymbol{h}_{[\mathrm{CLS}]}, \boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n, \boldsymbol{h}_{[\mathrm{SEP}]}\right\}$.

2) ENTITY TYPE EMBEDDING

In the ACE2005 dataset, each entity in the sentence is labeled with entity type information. Considering that entity type plays an important role in event trigger detection and classification, we introduce entity type features on the basis of word embeddings and hidden states sequence for the final event detection process. The embedded representation matrix of the entity type is denoted as $\boldsymbol{A}_t$ and is trained as a parameter of the model. Based on the entity type of each word, the corresponding entity type embeddings are noted as $\boldsymbol{T} = \left\{\boldsymbol{t}_{[\mathrm{PAD}]}, \boldsymbol{t}_1, \boldsymbol{t}_2, \cdots, \boldsymbol{t}_n, \boldsymbol{t}_{[\mathrm{PAD}]}\right\}$ by finding it from $\boldsymbol{A}_t$.

3) CLASSIFICATION LAYER

After the above steps, the word embeddings, entity type embeddings and hidden states of the event sentence are obtained. The word embedding $\boldsymbol{e}$, the entity type embedding $\boldsymbol{t}$ and the hidden state vector $\boldsymbol{h}$ of each word are spliced together to obtain the combined enhanced feature vector $\boldsymbol{e}^*$, as shown in (4):

$$\boldsymbol{e}^* = [\boldsymbol{e}, \boldsymbol{t}, \boldsymbol{h}] \tag{4}$$

The $\boldsymbol{e}^*$ is input to the fully connected neural network layer, and then the triggers are identified and classified by the output layer softmax function. The prediction probability $\boldsymbol{y}$ of each label is obtained, as shown in (5), where $\boldsymbol{W}_3$ denotes the weight parameter and $\boldsymbol{b}_3$ denotes the bias term.

$$\boldsymbol{y} = \mathrm{softmax}\left(\boldsymbol{W}_3 \times \boldsymbol{e}^* + \boldsymbol{b}_3\right) \tag{5}$$

D. LOSS FUNCTION

To train the event detector, we use cross-entropy loss function for optimization, which minimizes the loss value during training, and the loss function is shown in (6).Where y_{ij} is the predicted probability that the i-th word represents the j-th event type; $\hat{y}_{ij}$ is the truth-value representation, and $\hat{y}_{ij}$ is 1 if the true label of the i-th word is the j-th event type, otherwise it is 0; n is the number of words in the event sentence; and k is the number of all event types.

$$L_e = -\sum_{i=1}^{n}\sum_{j=1}^{k}\hat{y}_{ij}\log\left(y_{ij}\right) \tag{6}$$

In this paper, the event detection model ES4ED is trained using the joint training mode, which combines the losses of the two parts of the event sentence decider and event detector to obtain the final loss function, as shown in (7):

$$L = L_s + L_e \tag{7}$$

E. ALGORITHM DESCRIPTION

Algorithm 1 describes the detailed training process of ES4ED. Prior to training, the model parameters were initialized randomly (line 1). During the training, lines 3-21 are iterated epoch times, where epoch is set to 50. For each iteration we sample the $|TS|/bs$ batch text from the dataset. In the batch sample set BS, each sentence S obtains the word embedding matrix $\boldsymbol{E}$ by BERT, and the entity type embedding matrix $\boldsymbol{T}$ of S is obtained by finding the embedding representation matrix $\boldsymbol{A}_t$ of the entity type (lines 5 & 6). Next, $\boldsymbol{E}$ is fed into the event sentence decider to get the predicted label probability of the event sentence for S, and the loss of the event sentence decider module is calculated (lines 7-9). Temporarily write the sentence information predicted to be an event sentence into the set ES (lines 10 & 11). Further input word embeddings and entity type embeddings of event sentence into the event detector to get label predictions for triggers identification and classification (lines 13-16), and the loss is calculated and accumulated to the total loss of the current batch (lines 17 & 18). Finally, update all the parameters in the model (line 20).

IV. EXPERIMENT

A. DATASETS AND EXPERIMENTAL SETTINGS

To validate the effectiveness of the ES4ED, our experiments are conducted on the publicly available standard dataset ACE2005. The dataset contains 599 documents in English, mainly from broadcast news, broadcast conversation and telephone conversation. The ACE2005 dataset defines 8 major categories and 33 sub-categories of event types, as shown in Table 1, and we set the non-trigger word type to "None".

Algorithm 1 Detailed training process for ES4ED

Input: Training sample set TS, number of epochs $epoch$, batch size bs, batch sample set BS, entity type embedding dimension k, learning rate $\boldsymbol{\beta}$, the number of convolution window m, convolution window h

Output: $\boldsymbol{A}_t$, model parameters $= \{\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{b}_1, b_2, \boldsymbol{b}_3\}$

Initialize $\boldsymbol{A}_t$, $\{\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{b}_1, b_2, \boldsymbol{b}_3\}$
for i $= 1$ to $epoch$ **do**
  for BS in TS **do**
    for S in BS **do**
      $\boldsymbol{E} \leftarrow$ get the word embedding matrix of S via BERT
      $\boldsymbol{T} \leftarrow$ get the entity type embeddings of S via finding $\boldsymbol{A}_t$
      $\boldsymbol{s} = [\hat{c}^3, \hat{c}^4, \hat{c}^5] \leftarrow$ feed the $\boldsymbol{E}$ into the Convolutional Layer and then get its feature vector via Pooling Layer
      $p \leftarrow$ calculate the label prediction probability via Determination Layer
      $L_s \leftarrow$ compute the loss of the event sentence decider
      if $p > 0.5$ then
        $ES \leftarrow$ S $(\boldsymbol{E}, \boldsymbol{T})$
      end for
      for t in ES **do**
        $\boldsymbol{H} \leftarrow$ get the hidden state sequences of event sentence t via Bi-LSTM
        $\boldsymbol{e}^* = [\boldsymbol{e}, \boldsymbol{t}, \boldsymbol{h}] \leftarrow$ get the semantic information representation of each word
        $\boldsymbol{y} \leftarrow$ generate the prediction probability via Fully Connected Layer
        $L_e \leftarrow$ compute the loss of the event detector
        $L = L_s + L_e \leftarrow$ calculate the total loss of the model
      end for
    Update all the parameters
  end for
end for

TABLE 1. Event types contained in the English corpus in ACE2005.

Event Type	Event Subtype
Life	Be-Born, Die, Marry, Divorce, Injure
Transaction	Transfer-Ownership, Transfer-Money
Movement	Transport
Business	Start-Org, End-Org, Declare-Bankruptcy, Merge-Org
Conflict	Attack, Demonstrate
Personnel	Start-Position, End-Position, Nominate, Elect
Justice	Arrest-Jail, Release-Parole, Charge-Indict, Trial-Hearing, Sue, Convict, Sentence, Fine, Execute, Extradite, Acquit, Pardon, Appeal
Contact	Meet, Phone-Write

Following previous works, we employ the public toolkit ace2005-preprocessing[2] to preprocess ACE2005 dataset. In the preprocessing phase, the dataset is segmented according to the practice of mainstream research works. More precisely, we randomly selected 529 documents as the training set, 40 documents as the test set, and 30 documents as the validation set, with sentence as the basic unit for the experiments. The detailed statistics of the data are shown in Table 2.

[2]https://github.com/nlpcl-lab/ace2005-preprocessing

TABLE 2. Statistics on the English corpus in ACE2005. # means "the number of".

dataset	#sentence	#event sentence	# non-event sentence	#trigger	#argument
train	14582	3235	11347	4308	7982
test	873	340	533	426	937
validation	711	292	419	496	958
total	16166	3867	12299	5230	9877

Our model ES4ED is trained and tested on a server powered by Intel® Xeon® Gold 6226R, 128GB of running memory (RAM), and NVIDIA GeForce RTX 3090 Ti graphics. Under the development environment PyCharm, ES4ED is developed using Python programming language and deep learning framework PyTorch. During the training, we employ BERT to obtain the word embeddings, and the dimension is 768. We set the dimension of entity type embedding as 50 and the dimension of hidden state output from Bi-LSTM layer is 768. In convolution phase, the window widths of the three different convolutions are set to 3, 4, 5, and we use 100 convolutions for each different window width. The model is trained for 50 epochs with batch size 24 and we use Adam optimizer with learning rate 2e-5.

B. EVALUATION METRIC

For evaluation, we follow previous works and use Precision (P), Recall (R) and F1 score as the evaluation metrics to measure the effect of our model, as shown in (8)-(10):

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{8}$$

$$\mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{9}$$

$$\mathrm{F1} = \frac{2 \times \mathrm{P} \times \mathrm{R}}{\mathrm{P} + \mathrm{R}} \tag{10}$$

The event sentence decider is regarded as a binary classification task, we use Accuracy (ACC) as the evaluation metric of the event sentence decider. Accuracy indicates the proportion of samples with correct predictions to the total number of samples, and as shown in (11).

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{11}$$

Here, TP (True Positive) is the number of positive cases predicted as positive cases; FP (False Positive) is the number of negative cases predicted to be positive; FN (False Negative) is the number of positive cases predicted to be negative; TN (True Negative) is the number of negative cases predicted as negative cases.

C. MODEL COMPARISON

In this section, we compare ES4ED with the existing mainstream models to evaluate the strength and weaknesses of ES4ED. The main baseline models include:

1) DMCNN: Chen et al. [10] proposed a dynamic multi-pooling convolutional neural networks to extract sentence-level features and fuse them with lexical-level features for event extraction.
2) DBRNN: Sha et al. [12] proposed the recurrent neural networks based on dependency bridge for the joint extraction of events and arguments using the dependencies of words in sentence.
3) HBTNGMA: Chen et al. [23] proposed an annotation network structure based on a multi-level attention mechanism that uses features extracted from documents and sentences for event detection.
4) JMEE: Liu et al. [15] proposed an attention-based mechanism for graph convolutional neural networks that uses syntactic shortcut arcs to enhance information for joint extraction of events and arguments.
5) BGCN: Cheng et al. [24] proposed a trigger word detection model based on BERT and graph convolutional network to capture long-distance dependencies by introducing syntactic structures to achieve event trigger words detection.
6) TEXT2EVENT: Lu et al. [25] proposed a sequence-to-structure generative network for unified event extraction, and designed a constrained decoding algorithm and a curriculum learning algorithm.
7) MGRN-EE: Duan et al. [17] proposed a multi-grained ranking network for event extraction, where each subtask constructs a span representation from three granularities and uses two attention mechanisms to explore the association of event-event and event-entity.

TABLE 3. The comparison of experimental results of ES4ED with baseline models. Bold marks the highest score.

Model	TI (%)			TC (%)		
	P	*R*	F1	*P*	*R*	F1
DMCNN	80.4	67.7	73.5	75.6	63.6	69.1
DBRNN	-	-	-	74.1	69.8	71.9
HBTNGMA	-	-	-	77.9	69.1	73.3
JMEE	80.2	72.1	75.9	76.3	71.3	73.7
BGCN	**81.4**	74.1	77.6	**77.5**	72.4	74.2
TEXT2EVENT	-	-	-	69.6	74.4	71.9
MGRN-EE	-	-	-	71.2	77.6	74.3
ES4ED	73.8	**85.5**	**79.2**	71.3	**82.6**	**76.5**

Table 3 shows the experimental results of the ES4ED compared with the baseline models, and the results are shown: ES4ED achieves the best F1 score in both trigger identification (TI) and trigger classification (TC). Specifically, on the TI task, ES4ED improves F1 score by 1.6% compared with BGCN, which is currently the best performing model. On the TC task, ES4ED improves the F1 score by 2.3% compared to BGCN, and the F1 score increase of 2.2% compared with MGRN-EE, the current model with the best trigger classification effect. The reason why the ES4ED model can improve the event detection effect is, on the one hand, the use of the event sentence decider to separate out the event sentences first, which solves the problem of long-tailed distribution of events. On the other hand, it is the effective use of deep semantic features, which are used in the event classification process through fusion.

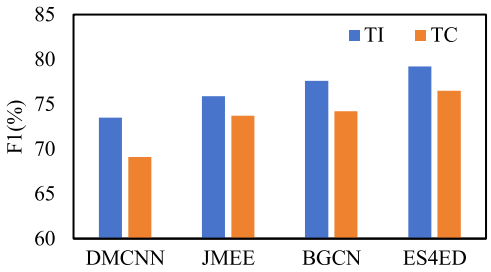

FIGURE 3. The comparison of F1 score of different models on TI and TC.

Fig. 3 shows the F1 score comparison of different models on TI and TC tasks, which is obvious: The F1 score of the different models on TC task is lower than that on TI task. In particular, the F1 score of DMCNN on TC task is 4.4% lower than that on TI task; the F1 score of JMEE on TC task is 2.2% lower than that on TI, and the F1 score of the ES4ED is also 2.7% different on these two tasks. The above results show that all kinds of models are easier to achieve good results on the TI. Through in-depth analysis of ACE2005 English corpus, it is found that the reason for this phenomenon is extremely the unbalanced distribution of the number of training samples for various event types in ACE2005. The statistical information on the number of training samples for each event type is shown in Fig. 4. It is easy to see that about 70% of the event types occur less than 100 times. Even the two most frequent event types, ''Attack'' and ''Transport'', differed by about 800 times.

For event types with a small number of training samples, it is difficult for the model to fully learn effective features of the event type during training, resulting in an overall decline in the performance of the event trigger classification task. Therefore, it is also one of the directions that we will concentrate on improving in our future work, and we still need to put in efforts.

D. ABLATION EXPERIMENT

To verify the effect of the event sentence decider on the event detection task, we compare the effect of ES4ED with ES4ED-det without the event sentence decider, and the results are shown in Table 4. As can be seen from Table 4: (1) On the task of TI, the P, R, and F1 score of ES4ED increased by 3.5%, 8.5% and 5.7%, respectively, compared with ES4ED-det; (2) on the task of TC, the P, R, and F1 score of ES4ED increased by 4.4%, 9.4% and 6.6%, respectively, compared with ES4ED-det. The above experimental results show that the effect of event detection can be greatly improved by filtering out the sentences that do not contain events and then detecting the event sentences.

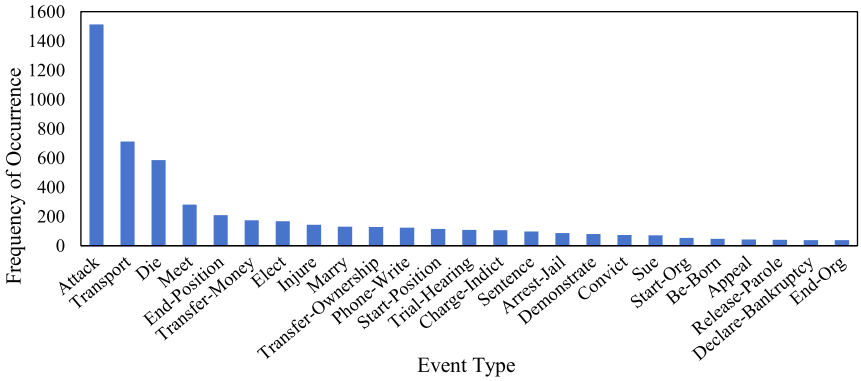

FIGURE 4. The horizontal coordinate is the event types in the dataset and the vertical coordinate is the number of occurrences of such events.

TABLE 4. The impact of event determiners on the effectiveness of event detection. ES4ED-det indicates ES4ED without the event sentence decider. Bold marks the highest score.

Model	TI (%)			TC (%)		
	P	*R*	F1	*P*	*R*	F1
ES4ED-det	70.3	77	73.5	66.9	73.2	69.9
ES4ED	**73.8**	**85.5**	**79.2**	**71.3**	**82.6**	**76.5**

To further explore the influence of the number of iterations epoch on the event detection effect during the training process, we plot the trends of the effect of ES4ED and ES4ED-det on TI and TC are plotted, and the results are shown in Fig. 5. In Fig. 5(a) and 5(b) show the F1 score changes of ES4ED and ES4ED-det with the increase of iteration epoch under TI and TC, respectively. It can be observed that the event sentence decider can improve the effect of the model on both TI and TC. As the number of iterations increases, the performance of the model shows an overall upward trend, and ES4ED achieves the best results on the trigger identification and classification when the epoch takes the value of 23.

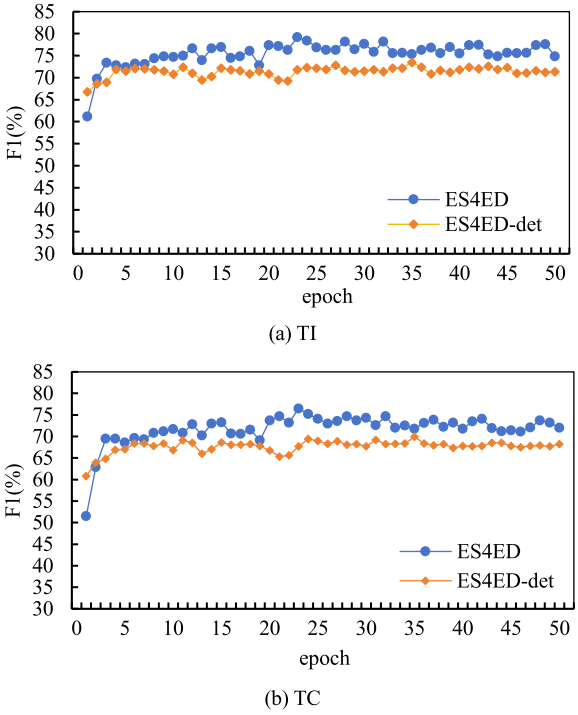

FIGURE 5. The effect of epoch on the detection effect during training. (a) shows the training effect of trigger identification (TI), and (b) shows the training effect of trigger classification (TC).

E. EFFECT OF CONVOLUTION WINDOW ON EVENT SENTENCE DECIDER

To explore the effect of convolution window on the event sentence decider, we separately utilize single convolution and three convolutions with different window width to extract feature. For the case of single convolution, the window width is set to 1/2/3/4. And for the case of three convolutions, the widths are set to (1,2,3)/ (2,3,4)/ (3,4,5)/ (4,5,6). The classification results of event sentences with different convolution window are shown in Fig. 6. It can be concluded that the effect of using multiple convolutions is generally better than that of single convolution. This is owning to multiple convolutions with different window width can extract features of different granularity from the text, and the sentence representation is better expressed by those features, which is conducive to improve the effect of event sentence decider. When the width of convolution window is set to (3,4,5), the event sentence decider gets the best performance, reaching 85.7%.

F. IMPACT OF MULTI-FEATURE FUSION ON EVENT DETECTION

To further explore the impact of the fusion of different features on the model performance, we conduct experiments with different fusion of three features: word embedding $\boldsymbol{e}$, entity type embedding $\boldsymbol{t}$, and hidden state vector $\boldsymbol{h}$. Fig. 7 shows the effect of TC under different features. In the figure, $e^{(1)}$ means only $\boldsymbol{e}$ as feature; $e^{(2)}$ denotes that $\boldsymbol{e}$ and $\boldsymbol{t}$ are used together as features to enhance the word representation; $e^{(3)}$ indicates that $\boldsymbol{e}$, $\boldsymbol{t}$, and $\boldsymbol{h}$ are combined as features to

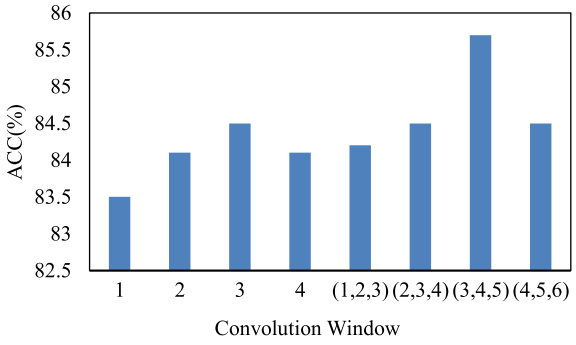

FIGURE 6. Classification performance with different convolution windows.

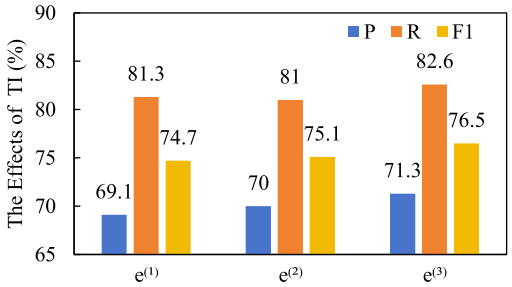

FIGURE 7. The performance of the model under different features. It shows the effect of trigger identification (TI).

enhance sentence representation. Analysis Fig. 7, it can be concluded that using $e^{(1)}$ as a feature for event detection is less effective because it is difficult to learn the semantic associations of contexts using only $\boldsymbol{e}$. The performance of $e^{(2)}$ improves marginally after enriching the feature by adding entity type information. With the addition of $\boldsymbol{h}$ containing contextual semantic information, $e^{(3)}$ combines both local and global feature information, at which point event detection is most effective. The experimental results show that multi-feature fusion can represent sentences more comprehensively, which effectively improves the effect of event detection.

V. CONCLUSION

In order to solve the problem of long-tailed distribution of events in data, we present a novel and effective event detection model based on a very simple idea. The event sentences are first extracted by binary classification, then the event sentences are detected and processed to solve the problem of long-tailed distribution of events. Finally, based on the pre-trained language module BERT and Bi-LSTM, the event trigger mention is identified and the corresponding type can also be classified. The effectiveness of our model is validated on the publicly available standard dataset ACE2005, and the experimental results on the event detection task demonstrate that ES4ED has significantly improved compared with the existing mainstream models. In our future work, document-level corpus features will be introduced into the event detection task in order to have a further improvement on the effect of event detection.

REFERENCES

[1] X. Ding, Z. Li, T. Liu, and K. Liao, "ELG: An event logic graph," 2019, *arXiv:1907.08015*.
[2] Z. Li, S. Zhao, X. Ding, and T. Liu, "EEG: Knowledge base for event evolutionary principles and patterns," in *Proc. Chin. Nat. Conf. Social Media Process.*, 2017, pp. 40–52.
[3] H. Han, Y. Fan, and X. Xu, "Multi-channel enhanced graph convolutional network for aspect-based sentiment analysis," *J. Electron. Inf. Technol.*, vol. 2023, pp. 1–11, Jan. 2022. [Online]. Available: http://kns.cnki.net/kcms/detail/11.4494.tn.20230928.1534.004.html
[4] S. Wu, M. Wang, Y. Li, D. Zhang, and Z. Wu, "Improving the applicability of knowledge-enhanced dialogue generation systems by using heterogeneous knowledge from multiple sources," in *Proc. 15th ACM Int. Conf. Web Search Data Mining*, Feb. 2022, pp. 1149–1157.
[5] A. Brandsen, S. Verberne, K. Lambers, and M. Wansleeben, "Can BERT dig it? Named entity recognition for information retrieval in the archaeology domain," *J. Comput. Cultural Heritage*, vol. 15, no. 3, pp. 1–18, Sep. 2022.
[6] C. Zhang, J. Han, Y. Zhang, and F. Li, "An interpretable free-text keystroke event sequence classification model," *J. Electron. Inf. Technol.*, vol. 15, no. 2, pp. 1–18, 2023.
[7] D. Ahn, "The stages of event extraction," in *Proc. Workshop Annotating Reasoning About Time Events (ARTE)*, 2006, pp. 1–8.
[8] HL. Chieu and H. Ng, "A maximum entropy approach to information extraction from semi-structured and free text," in *Proc. AAAI/IAAI*, 2002, pp. 786–791.
[9] Q. Li, H. Ji, and L. Huang, "Joint event extraction via structured prediction with global features," in *Proc. 51st. Snnu. Meet. Assoc. Comput. Ling.*, 2013, pp. 73–82.
[10] Y. Chen, L. Xu, L. Kang, D. Zeng, and J. Zhao, "Event extraction via dynamic multi-pooling convolutional neural networks," in *Proc. 53rd. Annu. Meet. Assoc. Comput. Ling.*, 2015, pp. 167–176.
[11] T. H. Nguyen, K. Cho, and R. Grishman, "Joint event extraction via recurrent neural networks," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2016, pp. 300–309.
[12] L. Sha, F. Qian, B. Chang, and Z. Sui, "Jointly extracting event triggers and arguments by dependency-bridge RNN and tensor-based argument interaction," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, 2018, pp. 1–8.
[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.
[14] A. Ramponi, B. Plank, and R. Lombardo, "Cross-domain evaluation of edge detection for biomedical event extraction," in *Proc. 12th Lang. Resour. Eval. Conf.*, 2020, pp. 1982–1989.
[15] X. Liu, Z. Luo, and H. Huang, "Jointly multiple events extraction via attention-based graph information aggregation," 2018, *arXiv:1809.09078*.
[16] S. Cui, B. Yu, T. Liu, Z. Zhang, X. Wang, and J. Shi, "Edge-enhanced graph convolution networks for event detection with syntactic relation," in *Proc. Findings Assoc. Comput. Linguistics (EMNLP)*, Nov. 2020, pp. 2329–2339.
[17] R. Duan, P. Guo, and D. Zhang, "Multi-grained ranking network for event extraction," *J. Phys., Conf. Ser.*, vol. 2347, no. 1, Sep. 2022, Art. no. 012023.
[18] J. Liu, Y. Chen, and J. Xu, "Saliency as evidence: Event detection with trigger saliency attribution," in *Proc. 60th. Annu. Meet. Assoc. Comput. Ling.*, May 2022, pp. 4573–4585.
[19] F. Li, W. Peng, Y. Chen, Q. Wang, L. Pan, Y. Lyu, and Y. Zhu, "Event extraction as multi-turn question answering," in *Proc. Findings Assoc. Comput. Linguistics (EMNLP)*, 2020, pp. 829–838. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.73
[20] I.-H. Hsu, K.-H. Huang, E. Boschee, S. Miller, P. Natarajan, K.-W. Chang, and N. Peng, "DEGREE: A data-efficient generation-based event extraction model," 2021, *arXiv:2108.12724*.
[21] S. Wang, M. Yu, S. Chang, L. Sun, and L. Huang, "Query and extract: Refining event extraction as type-oriented binary decoding," 2021, *arXiv:2110.07476*.
[22] X. Du and C. Cardie, "Event extraction by answering (Almost) natural questions," 2020, *arXiv:2004.13625*.
[23] Y. Chen, H. Yang, K. Liu, J. Zhao, and Y. Jia, "Collective event detection via a hierarchical and bias tagging networks with gated multi-level attention mechanisms," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 1267–1276.

[24] S. Cheng, W. Ge, Y. Wang, and J. Xu, "BGCN: Trigger detection based on BERT and graph convolution network," *J. Comput. Sci.*, vol. 48, no. 7, pp. 292–298, 2021.
[25] Y. Lu, H. Lin, J. Xu, X. Han, J. Tang, A. Li, L. Sun, M. Liao, and S. Chen, "Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction," 2021, *arXiv:2106.09232*.

JIZHAO ZHU received the Ph.D. degree in computer application technology from Northeastern University, in 2018. He is currently a Master's Supervisor with the School of Computer Science, Shenyang Aerospace University. His research interests include machine learning, knowledge graph, and natural language understanding.

HAONAN ZHAO received the B.S. degree in computer science and technology from Liaoning Petrochemical University, in 2020. She is currently pursuing the master's degree with the School of Computer Science, Shenyang Aerospace University. Her research interests include natural language processing, knowledge graph, and deep learning.

WENYU DUAN received the B.S. degree in electronic and information engineering from Qingdao University, in 2017. He is currently pursuing the master's degree with the School of Computer Science, Shenyang Aerospace University. His research interests include natural language processing, knowledge graph, and deep learning.

XINLONG PAN received the B.Sc. and M.Sc. degrees in radar engineering from the Air Force Radar Academy, China, in 2006 and 2010, respectively, and the Ph.D. degree in information and communication engineering from Naval Aeronautical University, China, in 2017. He is currently an Associate Professor with Naval Aeronautical University. His research interests include information retrieval, graph neural networks, and intelligent information processing.

CHUNLONG FAN received the Ph.D. degree in computer application technology from Northeastern University. He is currently a Professor and a Master's Supervisor with the School of Computer Science, Shenyang Aerospace University. His research interests include information physical systems and complex network analysis.