Neural Processing Letters (2024) 56:89
https://doi.org/10.1007/s11063-024-11570-8

DE3TC: Detecting Events with Effective Event Type Information and Context

Boyang Liu[1] · Guozheng Rao[1,3] · Xin Wang[1,3] · Li Zhang[2] · Qing Cong[1]

Accepted: 11 February 2024 / Published online: 6 March 2024
© The Author(s) 2024

Abstract
Event Detection (ED) is a crucial information extraction task that aims to identify the event triggers and classify them into predefined event types. However, most existing methods did not perform well when processing events with implicit triggers. And most methods considered ED as a sentence-level task, lacking effective context for event semantics. Moreover, how to maintain good performance under low resource conditions still needs further study. To address these problems, we propose a novel end-to-end ED model called DE3TC, which **D**etects **E**vents with **E**ffective **E**vent **T**ype Information and **C**ontext. We construct an event type-specific Clue to capture the interaction between event type name and trigger words, providing event type information for implicit triggers. For accessing the effective context of event semantics for sentence-level ED, we consider the correlations between types and select similar types' descriptions as context. With contextualized representation from a contextual encoder, DE3TC learns the event type information for all events including implicit ones. And it performs sentence-level ED efficiently with effective contexts. The empirical results on ACE 2005 and MAVEN datasets show that: (i) DE3TC obtains state-of-the-art performance compared with previous methods. (ii) DE3TC is also excelled under low-resource conditions.

B. Liu and L. Zhang have contributed equally to this work

✉ Xin Wang
wangx@tju.edu.cn

Boyang Liu
lby_014@tju.edu.cn

Guozheng Rao
rgz@tju.edu.cn

Li Zhang
zhangli2006@tust.edu.cn

Qing Cong
chf@tju.edu.cn

1 College of Intelligence and Computing, Tianjin University, Tianjin 300350, China

2 School of Economics and Management, Tianjin University of Science and Technology, Tianjin 300457, China

3 Tianjin Key Laboratory of Cognitive Computing and Applications, Tianjin 300350, China

Keywords Event detection · Event type information · Effective context · Low-resource learning

1 Introduction

Event Detection (ED) is an essential yet challenging information extraction task in the field of Natural Language Processing. An event is identified by a word or a phrase called event trigger which most represents that event. Given an input text, ED aims to identify the event triggers and classify them into predefined event types. For instance, in the input sentence "*Stewart's 1979 marriage to Alana Hamilton lasted five years and produced two children.*", an ED model needs to recognize the word "*marriage*" as an event trigger and predict its event type as "*Life.Marry*".

Early ED methods explored statistical information in the training sets and used pattern-based methods [1–3]. Due to the excellent performance of neural network in the field of natural language processing, many ED models used various neural networks to extract contextual semantic features of events [4–6], such as Convolutional Neural Networks (CNN) [4], Recurrent Neural Network (RNN) [5], and Graph Convolutional Network (GCN) [6]. Recently, with the development of the Pre-trained Language Models [7–10] based on the transformer [11] architecture, many powerful ED models have emerged, which better understand the semantics of events in the context and have made significant improvements [12–14]. Some of them regarded ED tasks as question answering (QA)/machine reading comprehension tasks and detected events by finding answers to pre-defined questions [14–16]. For instance, Du et al. [14] formulated ED as a QA task and designed several queries for event triggers. Some generation-based methods manually defined templates or output formats to accomplish event detection [17–19]. They utilized event type information and achieved good performance. Lu et al. [18] designed a sequence-to-structure network and generated different structures for different event types. Hsu et al. [19] proposed a generation-based method and manually designed a prompt containing event type descriptions to guide the process of event detection. According to Liu et al. [20], to better leverage the capabilities of pre-trained language models (LM), various methods reformulate downstream tasks, making them more akin to those solved during the original LM training with the help of a textual prompt. The prompt is derived from the original input using a designed template and serves as the input for the LM. Prompt-based methods reformulate the target task into a generative task, attempting to learn a LM to perform the original task, reducing or obviating the need for large supervised datasets. This process requires additional labor for both task reformulation and template design.

There are still three problems as follows that have not been solved:

(1) How to detect event with implicit trigger simply but efficiently?

In real-life situations, events within the text may often be implicit. In these cases, trigger words do not explicitly convey the semantics of events. For example, as shown in Fig. 1 (Example 1), we can easily tell that the word "*marriage*" is the trigger for the event type "*Life.Marry*". However, in Example 2, it is challenging to determine that "*deployed*" is the trigger word for the event type "*Movement.Transport*". Therefore, we need event type information to identify such trigger words.

Most existing methods were unable to handle this situation satisfactorily. The interaction between event types and trigger words provides event type information, which greatly aids in identifying implicit trigger words. Some excellent methods based on QA or machine

Example 1:
Sentence: Stewart's 1979 **marriage** to Alana Hamilton lasted five years and produced two children.
Event type: Life.Marry
Trigger: marriage

Example 2:
Sentence: They also **deployed** along the border with Israel.
Event type: Movement.Transport
Trigger: deployed

Fig. 1 Two examples of sentence-level event detection from ACE 2005 dataset

reading comprehension only provided a keyword as a query for the event type, for example, "[EVENT]". However, these methods failed to obtain event type information. Some prompt-based methods designed different templates for each event type. They may capture the event type information to some extent. But they required a significant amount of manual effort, which is not feasible in real-life applications.

(2) How to access effective context of event semantic for sentence-level ED?

Context is essential for semantic understanding. The sentence-level event detection task often lacks effective context. It is difficult to identify the trigger words based solely on the semantics of a single sentence. As shown in Fig. 1 (Example 2), the sentence lacks effective contextual information of event semantic. If we provide more event semantics for "*Movement.Transport*", the model can identify "*deployed*" as the trigger word more easily.

Furthermore, most existing methods ignored the correlations between similar event types, which provide valuable contextual information of event semantic. For example, ("*Transfer.Money*", "*Transfer.Ownership*"), ("*Execute*", "*Sentence*") are two pairs of similar event types, and there are prominent correlations between them. The correlations between similar event types can provide effective context and help the model learn more semantic knowledge [21].

(3) How to achieve good performance under low resource conditions?

Most existing ED methods followed the supervised learning paradigm and relied on a large number of high-quality annotated texts. When training data is insufficient, their performance becomes suboptimal. In practical applications, it is very expensive to obtain high-quality annotation data. Therefore, how to design an efficient ED model that can achieve good performance with only a small amount of annotated texts has become a key challenge.

To address all these problems, we propose a novel model named **DE3TC**, which **D**etects **E**vents with **E**ffective **E**vent **T**ype Information and **C**ontext in an end-to-end way, as shown in Fig. 2. We design a **Clue** for each event type to capture interaction between event types and trigger words. Distinguished from prompt, our Clue is constructed automatically and does not require additional human labor without reformulating the task. These features of Clue make our approach more practical in real-world applications. DE3TC obtain event type information simply but efficiently by capturing interaction between event types and trigger words with Clue. And it accesses effective context of event semantics by considering the correlations between similar event types.

Specifically, we designed an Event Type Information Constructor to construct Clues. The effective context of event semantics can be accessed by using a Context Selector consid-

ering the correlations between event types. The selector finds similar event types and adds their descriptions to the input sequence as contexts. This process is automated and exhibits good generalization. Finally, a Clue, the corresponding context and the given sentence form the event semantic modeling sequence. Such composition enables the sequence to contain type information and effective context of event semantics. Then we use the the Contextual Encoder to obtain contextualized representations of the event semantic modeling sequence. With event type information and effective contexts, DE3TC has powerful event detection ability and performs well under low-resource conditions. Details will be explained in Sect. 3. Finally, DE3TC performs ED efficiently in an end-to-end way and achieves state-of-the-art performance. Our contributions can be summarized as follows:

- We propose DE3TC, an efficient end-to-end ED model obtaining event type information with Clue by capturing interaction between event types and trigger words simply but efficiently with an Event Type Information Constructor.
- We design a Context Selector to obtain effective contexts by selecting similar event types with correlations.
- DE3TC with event type information and effective contexts achieves high performance under low-resource conditions.
- The experimental results on ACE 2005 and CASIE dataset demonstrate the strong performance (state-of-the-art) of DE3TC.

2 Related Work

2.1 Event Detection

Event Detection (ED) is an information extraction task that has been studied for a long time. The traditional methods collected statistical information from training sets as the knowledge source of the ED model and used pattern-based methods [1–3]. For example, Li et al. [2] discovered frequent patterns and aggregated strongly correlated frequent patterns together to perform ED task. Qin et al. [3] proposed feature-based event filtering to study segment-based news event detection. These methods were sensitive to variations in the text, making it difficult to generalize to other datasets with different statistical characteristics.

With the development and introduction of deep learning technology, many ED methods based on neural networks extracted contextual semantic features from each text via end-to-end architectures [4–6]. For instance, Chen et al. [4] used a convolutional neural network (CNN) to automatically extract lexical-level and sentence-level features. Nguyen et al. [5] proposed to do event extraction in a joint framework with bidirectional recurrent neural networks (RNN). Cui et al. [6] proposed a novel architecture named Edge-Enhanced Graph Convolution Networks (EE-GCN), which simultaneously exploited syntactic structure and typed dependency label information to perform ED. Najafipour et al. [22] devise a neural network-based temporal-textual framework for linking authors of short-text contents to overcome the challenge of temporal skew in textual content. Liu et al. [23] proposed a time-aware entity alignment (TEA) model to discover the entity evolving behaviour by exploring the time contexts in knowledge graphs. Feng et al. [24] proposed an interlayer feature fusion-based heterogeneous graph neural network model to enhance the representation of the original input features. Lyu et al. [25] proposed a bidirectional lattice graph attention network (BiLGAT) to fully utilize the advantages of pretrained language model and lexicon features. However,

all these methods had their own limitations in feature processing, such as CNN cannot learn long distance dependencies. And they had limited understanding of context.

Recently, many studies applied powerful pre-trained language models based on the transformer [11] architecture to better comprehend contexts and achieve great improvements [12, 13]. For example, Du et al. [14] introduced a new paradigm for event extraction by formulating it as a question answering (QA) task. Based on BERT (Bidirectional Encoder Representations for Transformers) [7] model, they designed a variety of queries and achieved good results on event extraction by finding the corresponding answers to queries. But they only use simple query which is a single word to perform event detection. Saaki et al. [26] proposed a method with time-sensitive value-wise transformer to find the most suitable individual to answer a question, which track user textual-temporal behavioral patterns via an infinite continuous-time module. Lu et al. [18] designed a generation-based sequence-to-structure network for unified event extraction. They first converted event records into a labeled tree and then linearized it into a token sequence via depth-first traversal. They need to construct a tree structure for each example, which is labor-intensive. Neither of these two methods captured the interaction between event types and trigger words, resulting in suboptimal event detection performance. Hsu et al. [19] proposed a generation-based method called DEGREE and manually designed a prompt containing event type descriptions and event keywords. DEGREE got excellent performance on event extraction, especially under low-resource conditions. But the design process of its prompt costs a lot of manual work, which is unacceptable in real-world applications. Hosseini et al. [27] propose a probabilistic generative model called multi-aspect time-related influence (MATI) to consider the fact that time includes numerous granular slots. Liu et al. [28] proposed a generative template-based event extraction method with a dynamic prefix (GTEE-DYNPREF). GTEE-DYNPREF learned a context-specific prefix for each context by integrating context information with type-specific prefixes. It computed a prefix vector each time, consuming a lot of time, and did not select context for each event type. Sheng et al. [29] performed ED task by integrating type-level and instance-level correlations. They learned more informative type representations and leveraged co-occurrence events as remarkable evidence in prediction. It took into account the event type-level dependency, but did not provide an effective context for event semantics.

In this paper, we propose DE3TC that captures event type information in a simple but efficient way (Clue). And our method considers type-level correlations as the context of event semantics. Compared with previous methods, our approach is not based on specific statistical information, demonstrating strong generalization. Specifically, we use Event Type Information Constructor to construct Clues to capture interaction between event types and trigger words simply but efficiently for event type information. Our Clues accurately represent event type information, and they are generated automatically. Unlike existing methods, Clues are type-specific and constructed without the need for human labor to design them based on different event types. The Context Selector in our method provides an effective context for event semantics in sentence-level event detection. Distinguishing from existing methods that recognize an event based only on a sentence, our method provides more event semantic information through context.

2.2 Low-Resource Event Detection

Most of the existing methods relied on a large number of annotated data for events. However, in real-world applications, obtaining high-quality annotated data is very expensive.

Some previous ED methods performed well in low-resource conditions [18, 19]. Lu et al. [18] proposed a generation-based sequen-ce-to-structure network that had the ability to learn in a low-resource condition. Hsu et al. [19] used manually designed templates to achieve good performance under low-resource conditions. However, different templates should be specially designed for different event types, which requires many human resources. We design a simple Clue that does not need lots of manual work. We use Clues to capture event type information and access the context of event semantics so that our model is able to perform well under low-resource conditions.

3 Methodology

In this section, we present the general framework of DE3TC. We first provide an overview of the framework (Sect. 3.1). Then we provide detailed descriptions of the three main components of the model, Event Type Information Constructor (Sect. 3.2), Context selector (Sect. 3.3), and Contextual Encoder (Sect. 3.4). We also give some details of the inference (Sect. 3.5).

3.1 Framework Overview

As illustrated in Fig. 2, we first construct a set of Clues (see Sect. 3.2 for details) from the train set for all event types to express event semantics by Event Type Information Constructor. The Clues capture the interaction between the event types and the trigger words simply but efficiently. This mechanism provides event type information for cases with implicit trigger words.

Furthermore, we design a Context Selector to access the context of the event semantics. The selector captures correlations between similar event types and utilizes their descriptions as context, addressing the deficiency of effective context in sentence-level event detection.

Then we concatenate the input sentence with its Clue and the context of the event semantics as the event semantic modeling sequence. Finally, we put the sequence into the Contextual Encoder to obtain the context representation. The format of the event semantic modeling sequence is as follows:

$$\text{[CLS]}\ \langle Clue \rangle\ \text{[SEP]}\ \langle Sentence \rangle\ \text{[SEP]}\ \langle Context \rangle\ \text{[SEP]},$$

where [CLS] is special classification token, [SEP] is the special token to denote separation. $\langle Clue \rangle$, $\langle Sentence \rangle$, and $\langle Context \rangle$ are the tokenized sequences. Finally, we accomplish event detection by calculating the probability of each word as a trigger word for a specific event type and get the result of event prediction.

3.2 Event Type Information Constructor

In text with implicit events, trigger words do not explicitly express event semantics, which is common in real-life situations as illustrated in Fig. 1 (Example 2). The semantic information of event types is crucial for the event detection model. The interaction between event types and trigger words provides semantic information of event types. The existing methods ignored such interaction and did not have event-specific information or require a lot of manual work. To capture the interaction between event types and trigger words for event type information,

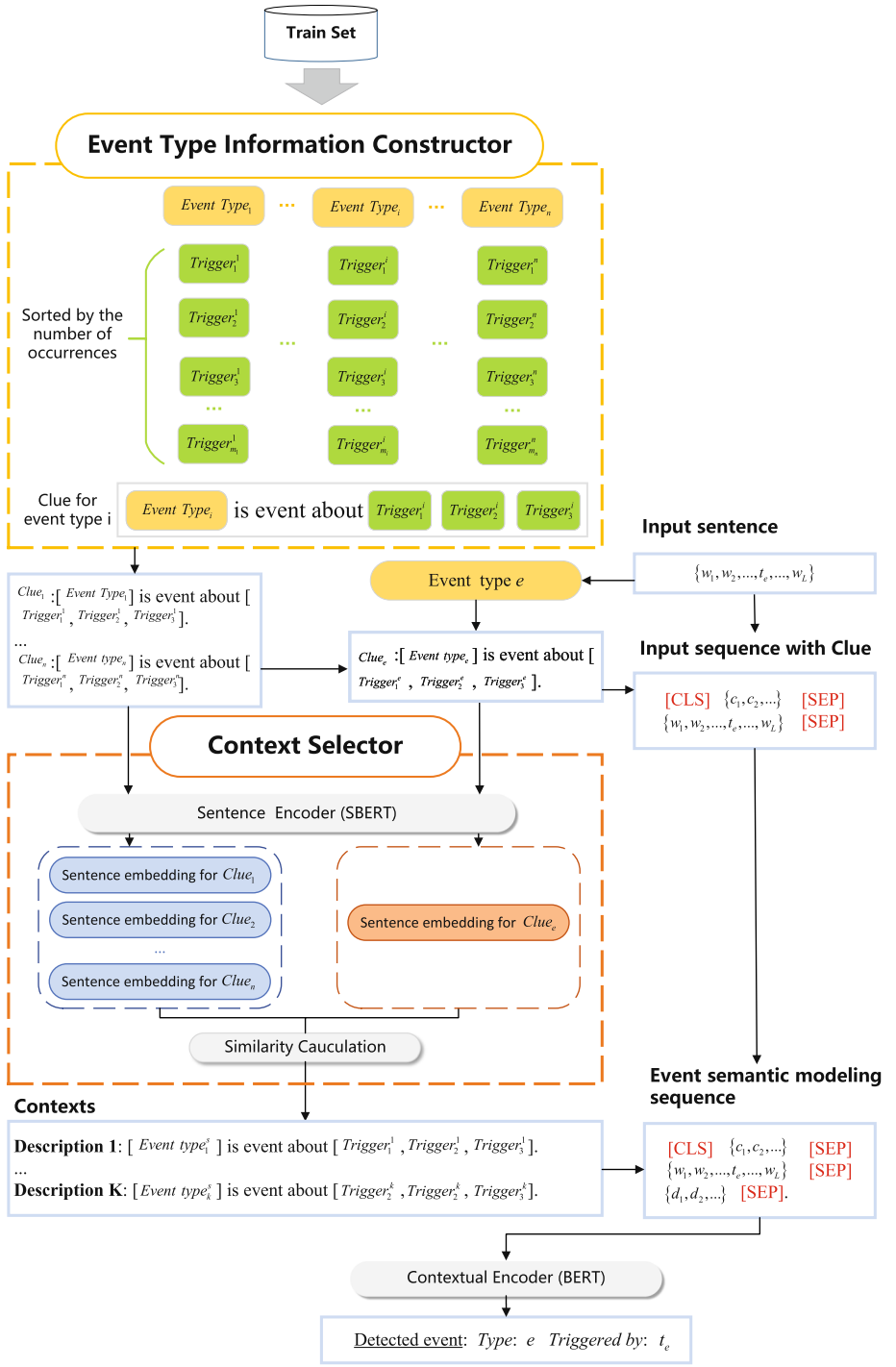


Fig. 2 Overall framework of DE3TC

we design a novel Event Type Information Constructor. The constructor is utilized to construct an event "**Clue**" in a simple but efficient manner.

A Clue is a sentence containing the event type name and a keyword list related to the event. We design a formwork where the model fills in the event type names and the corresponding common trigger words as the keyword list based on the input samples. By feeding the Clue into a contextual encoder, which is bidirectional, model captures the interaction between event types and trigger words.

As shown in Fig. 2, the Event Type Information Constructor constructs a Clue for every event type. In practice, we collected all the trigger words in the train set for each event type and selected the three most frequent ones as the keyword list, as shown in Fig. 2. Then we form a complete Clue:

$$\langle Type\ name \rangle \text{ is an event about } \langle Keywords\ list \rangle,$$

where $\langle Type\ name \rangle$ and $\langle Keywords\ list \rangle$ are special tokens respectively serving as the placeholders of the event type name contained in the sentence and the keyword list corresponding to the event type. In training, we replace $\langle Type\ name \rangle$ and $\langle Keywords\ list \rangle$ with event type name and corresponding keyword list. For example, the Clue of *Movement.Transport* event is "*Movement.Transport is an event about travel go move*".

Clues have the following advantages: (1) Clues are type-specific and contain event type names and trigger words. Compared with the questions in QA-based methods [14, 15], which were mostly one-word. Clues capture interaction between event types and trigger words and contain more accurate event type information. (2) Compared with prompt-based methods [19], whose prompt requires a lot of manual work, Clues are automatically generated by the script we wrote ourselves. The script automatically generates the corresponding Clue with the given formwork. It is simple to implement and easier to apply in practical applications.

Finally, each event type will be represented with a corresponding Clue, which contains rich event type information. Given an input sentence of length L, $\{w_1, w_2, ...w_L\}$, where w_i indicates the i-th token in the sentence, the corresponding $Clue_e$ is retrieved from the Clues which are generated by Event Type Information Constructor. Then we get input sequence with Clue as shown in Fig. 2, where $\{c_1, c_2, ...\}$ indicates token list of $Clue_e$.

3.3 Context Selector

Context plays an essential role in the understanding of event semantics, especially in sentence-level ED. As shown in Fig. 1 (Example 2), it is difficult to identify the word "*deployed*" as the trigger of a "*Movement.Transport*" event based solely on the semantics of the single sentence. On the other hand, there are many similar event types in ED tasks. For example, ("*Transfer.Money*", "*Transfer.Ownership*"), they both mean to transfer things. Semantic correlations exist between such event types and can be used as context to help event detection. So we select similar event types and regard them as the context of event semantics.

Only event type names cannot accurately express type semantics. So we need to use event type descriptions when calculating the similarity between event types. The previous section constructed a set of Clues for all event types. We use the set of Clues as the descriptions of event types and calculate the type similarity between descriptions to find similar types. As shown in Fig. 2, after we get $Clue_e$ for the input sentence, we input $Clue_e$ and all other Clues to the Context Selector as event descriptions. Context Selector first maps the descriptions into dense representations using a sentence encoder. Then a similarity calculation module is used to get the K most similar event descriptions to the $Clue_e$.

A direct method for obtaining sentence representations is to use contextual encoder, for example, BERT [7], to obtain the representation of all tokens in a sentence, and then use the mean-pooled result of all token representations. However, such a method derives sentence representations from token representations, which may introduce biases in sentence representations. We use Sentence-BERT (SBERT) [30] as our sentence encoder to get the sentence embedding of all event type descriptions. SBERT is a modification of the BERT model that uses siamese and triplet network structures to obtain semantically meaningful embeddings for sentences. And it is trained on Semantic Textual Similarity (STS) data, making SBERT suitable for obtaining sentence representations and their textual similarity.

Then we calculate the cosine similarity between each description and all other descriptions and get a similarity score (range in $[-1, 1]$). When the score is less than 0, there is no correlation between the two sentences. Otherwise, the two sentences are positively correlated when the score is greater than 0. The higher the score, the more significant the correlation between the two sentences.

Then for each event type, we sort all other types according to the similarity scores from the largest to the smallest to obtain a list of similar type descriptions. Then we choose the top K type descriptions as the final result of the Similar Type Selector, where K is a hyperparameter.

Finally, we have K type descriptions as our effective context of event semantics to make up event semantic modeling sequence with Clue and contexts as shown in Fig. 2, where $\{d_1, d_2, ...\}$ indicates token list of contexts.

3.4 Contextual Encoder

We obtain the event semantic modeling sequence in the format described in Sect. 3.1. Then we use a pre-trained BERT encoder to encode the event semantic modeling sequence. The model is trained on the Masked Language Model and Next Sentence Prediction tasks. BERT is designed to pre-train deep bidirectional representations from the unlabeled text by joint conditioning on both the left and the right context in all layers. The pre-trained BERT model can be fine-tuned with just one additional output layer to create advanced models for many natural language processing tasks.

For the tokenized event semantic modeling sequence with N tokens $\{w_i\}_{i=1}^{N}$, we have:

$$E = \{e_i\}_{i=1}^{N}, \tag{1}$$

$$\{e_i\}_{i=1}^{N} = BERT(\{w_i\}_{i=1}^{N}), \tag{2}$$

where E is the sequence of contextualized representations, e_i is the contextualized representation of input token w_i from the pre-trained language model. The function $BERT()$ represents a pre-trained BERT model. The output layer of the model predicts the event type for each token in the sentence. More specifically, we introduce a new parameter matrix $W \in \mathbb{R}^{H \times T}$ where H is the hidden size of the transformer, and T is the number of event types plus one (for non-trigger tokens). Softmax normalization is applied across the T types to produce the probability distribution across the event types P:

$$P = softmax(EW) \in \mathbb{R}^{T} \times N. \tag{3}$$

At test time, to obtain the type for each token $\{e_i\}_{i=1}^{N}$, we simply apply argmax to P.

During training, we fix the parameters of SBERT, so the similar type selector's parameters are not updated during the training. We use cross entropy between the prediction and golden labels as our training loss to fine-tune a pre-trained BERT model.

Table 1 Statistics for the ACE 2005 dataset and CASIE dataset

Dataset	Split	# Document	# Sentence	# Event
ACE 2005	Train	529	17,172	4202
	Dev	30	923	450
	Test	40	832	403
CASIE	Train	697	11,189	5044
	Dev	100	1778	2138
	Test	200	3208	1123

3.5 Inference

At test time, the event type information is unavailable to the model, so we calculate the interaction results between each test sample and all event types. Specifically, for each test sample, we take Clues of all event types and their corresponding context as input. The trigger word with the highest probability and the corresponding event type are used as the prediction result. For example, for input x, we construct M sequences (M is the number of event types in the dataset):

$$[\text{CLS}]\ Clue_i\ [\text{SEP}]\ x\ [\text{SEP}]\ Context_i\ [\text{SEP}],$$

where $Clue_i$ and $Context_i$ are the Clue and context corresponding to the ith event type, $i \in (1, M)$.

4 Experiments

4.1 Dataset and Evaluation Metric

We conduct our experiments on the widely-used ACE 2005 dataset[1] and CASIE dataset [31]. ACE 2005 dataset contains 599 documents with 33 event types. It contains documents crawled between year 2003 and 2005 from a variety of areas such as newswire, weblogs, broadcast conversations and broadcast news. CASIE dataset comprises a corpus of 1000 English news articles from 2017 to 2019, annotated with rich, event-based information. The dataset covers five event types, including cyberattack and vulnerability-related events. For ACE 2005 dataset, we preprocess the data and split the dataset as training, developing, and testing sets according to previous works [13]. For CASIE, we first remove three incomplete annotated documents and split the dataset as training, developing, and testing sets according to previous works [32]. Statistics of the data splits for ACE 2005 and CASIE are shown in Table 1.

As for evaluation in ACE 2005 dataset, we compare our model with previous works on two subtasks: (1) Trigger Identification (Tri-ID): an event trigger is correctly identified if its offsets match those of a gold-standard trigger; (2) Trigger Classification (Tri-CLS): an event trigger is correctly classified only if it is correctly identified and classified to the corresponding event type. Although our model DE3TC is an end-to-end model, which processes trigger word identification and classification at the same time, we still report the trigger identification results to compare to prior methods. As for evaluation in CASIE dataset, we report the

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

results for the trigger classification subtask for comparison with the baseline models. We report the official precision (P), recall (R) and F1 scores (F1) for evaluation and provide a comprehensive overview of three metrics as follow:

First we define the four fundamental components of evaluation: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN).

(1) True Positive (TP): The instances that are correctly predicted as positive by the model.
(2) False Positive (FP): The instances that are incorrectly predicted as positive by the model.
(3) True Negative (TN): The instances that are correctly predicted as negative by the model.
(4) False Negative (FN): The instances that are incorrectly predicted as negative by the model.

Then we introduce the key evaluation metrics: Precision, Recall, and F1 Score.

Precision: Precision measures the accuracy of positive predictions made by a model. It is calculated as the ratio of true positive predictions to the sum of true positive and false positive predictions. Precision is particularly valuable in scenarios where the cost of false positives is high. Mathematically, Precision is defined as:

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{4}$$

Recall: Recall gauges the ability of a model to capture all instances of the positive class within the dataset. It is computed as the ratio of true positive predictions to the sum of true positives and false negatives. Recall is crucial in situations where missing positive instances is costly. Mathematically, Recall is defined as:

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{5}$$

F1 Score: The F1 Score is a harmonic mean of Precision and Recall, providing a balanced measure that considers both false positives and false negatives. It is particularly useful when there is an uneven class distribution. The F1 Score reaches its maximum at 1 (perfect precision and recall) and its minimum at 0. Mathematically, F1 Score is defined as:

$$\text{F1 Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}. \tag{6}$$

4.2 Implementation Details

For the BERT encoder in DE3TC, we adopt bert-base-uncased in huggingface,[2] which has 768 hidden embedding dimensions and 12 attention layers, and each layer has 12 heads. We set the hyperparameter K to 2, which means the similar type selector selects the top 2 similar event type descriptions. The learning rate is 4e−05, the batch size is 8. We train our models with Adam optimizer [33] and 10% warming-up steps. Our experiments run on one NVIDIA Geforce RTX 3090.

4.3 Baseline Methods

On ACE 2005, we compare our model with the following methods:

(1) **EE-GCN** [6], which simultaneously exploited syntactic structure and typed dependency label information to perform ED;

[2] https://github.com/huggingface/transformers.

(2) **OneIE** [12], an end-to-end information extraction system which employs global feature;
(3) **BERT_QA** [14], a QA-based method that viewed event extraction tasks as a sequence of extractive question answering problems;
(4) **RCEE_ER** [15], which formulated ED task as a machine reading comprehension problem;
(5) **MQAEE** [16], which converted event extraction to a series of question answering problems;
(6) **BART-GEN** [17], a template-based conditional generation method;
(7) **Text2Event** [18], a sequence-to-structure generation model that converted the input passage to a tree-like event structure;
(8) **TANL** [34], which treated ED tasks as translation tasks between augmented natural languages;
(9) S^2**-JDN** [35], which introduced a set of statistical features from word-event co-occurrence frequencies to cooperate with the contextual features;
(10) **DEGREE** [19], a data-efficient model that formulated event extraction as a conditional generation problem;
(11) **GTEE-DYNPREF** [28], a generative template-based event extraction method with dynamic prefix by integrating context information with type-specific prefixes to learn a context-specific prefix for each context;
(12) **CorED-BERT** [29], which simultaneously incorporated both the type-level and instance-level event correlations and reached the current state-of-the-art performance on ED. We use its BERT encoder version as a comparison.

On CASIE, we compare our model with Text2Event and **UIE** [32]. UIE is a unified text-to-structure generation framework, which can universally model different IE tasks, adaptively generate targeted structures, and collaboratively learn general IE abilities from different knowledge sources.

4.4 Main Results

The evaluation results are shown in Tables 2 and 3. We can observe that: (1) Our DE3TC model outperforms all the baselines and achieves state-of-the-art performance on Tri-ID and Tri-CLS on ACE 2005. Significantly, DE3TC improves by 1.98% in F1 score of Tri-CLS over the best performance baselines, CorED-BERT. On the CASIE dataset, our method significantly outperforms baseline methods, achieving a 10.78% improvement in F1 score compared to Text2Event.

(2) Compared with DEGREE on ACE 2005, the event extraction method using prompts with manually designed type descriptions and keywords, our model outperforms it significantly. It achieves an increase of 10.98% in F1 score of Tri-CLS. Our combination of the Clue and context is better than the prompt in DEGREE. It can capture event type information in the text and make the model more sensitive to implicit trigger words.

(3) The recall of our method on Tri-CLS is slightly lower than GTEE-DYNPREF in Table 2, but the precision is far higher than it. It shows that GTEE-DYNPREF integrating context information with type-specific prefixes captures more events. However, its low precision shows that it also captures many wrong events. Our method is good at balancing precision and recall to achieve better comprehensive performance.

Table 2 Precision (P), Recall (R), and F1 scores (F1) of Trigger Identification (Tri-ID) and Trigger Classification (Tri-CLS) on ACE 2005 dataset

Model	Tri-ID			Tri-CLS		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
TANL	–	–	72.90	–	–	68.40
OneIE*	73.04	74.76	73.89	70.05	71.70	70.86
BART-Gen	72.69	76.12	74.36	69.53	72.81	71.13
TEXT2EVENT	–	–	–	69.60	74.40	71.90
BERT_QA	74.29	77.42	75.82	71.12	73.70	72.39
DEGREE*	70.40	82.63	76.03	66.46	79.65	72.46
GTEE-DYNPREF	–	–	–	63.70	**84.40**	72.60
MQAEE	–	–	77.40	–	–	73.80
RCEE_ER	–	–	–	75.60	74.20	74.90
EE-GCN	–	–	–	76.70	78.60	77.60
S^2-JDN	78.62	**85.60**	81.96	76.27	83.04	79.51
CorED-BERT	–	–	–	79.90	81.70	81.20
DE3TC	**85.25**	84.61	**84.93**	**83.5**	82.87	**83.18**

The best results in each column of the table are shown in bold. *Marks results produced with official implementation

Table 3 Precision (P), Recall (R), and F1 scores (F1) of Trigger Classification (Tri-CLS) on CASIE dataset

Model	P (%)	R (%)	F1 (%)
TEXT2EVENT	–	–	67.51
UIE*	68.32	70.05	69.17
DE3TC	**78.97**	**80.96**	**79.95**

The best results in each column of the table are shown in bold. *Marks results produced with official implementation

4.5 Low-Resource Experiments Results

Most existing supervised ED methods relied on a large number of annotated texts. However, in practice, it is very expensive to obtain a large number of high-quality event annotations. Therefore, it is particularly crucial to study the ED capability of models under low data resources.

In order to study the quick learning ability of DE3TC under low-resource conditions, we conduct experiments on ACE 2005 dataset under three different low-resource settings so that the model only learns 10/20/30% of the training data. We split the data according to the DEGREE [19]. We compare our model with **OneIE** [12], **BERT_QA** [14], **Text2Event** [18], **TANL** [34] and **DEGREE** [19] as mentioned in the previous section, and report the F1 scores of Tri-CLS. The experimental results are shown in Table 4.

From Table 4, we observe that DE3TC performs best, significantly outperforms other baselines on 20% and 30% of training data, and gets a competitive result on 10%. With only 10% of the training data, DE3TC performs poorly. This indicates that our method needs to capture the interactions of trigger words and event types well with minimal training samples, resulting in poorer performance. However, results on 20% and 30% of training data demonstrate that our model has a strong ability to learn under relatively low resource

Table 4 F1 scores(%) of Tri-CLS with different proportions of training data

Model	10%	20%	30%
Text2Event	47.00	55.60	60.70
BERT_QA	50.10	61.50	61.30
TANL	54.80	61.80	61.60
OneIE	61.50	67.60	67.40
DEGREE	**65.80**	68.30	68.20
DE3TC	48.90	**70.44**	**73.18**

The results of the baseline models are all from the DEGREE [19]. The best results in each column of the table are shown in bold

Table 5 Experimental results of ablation study on ACE 2005 dataset

Model	Tri-ID			Tri-CLS		
	P (%)	R (%)	F1 (%)	P (%)	R (%)	F1 (%)
DE3TC	**85.25**	84.61	**84.93**	**83.5**	82.87	**83.18**
DE3TC w/o Clue	81.52	82.13	81.82	72.41	72.95	72.68
DE3TC w/o Context	82.54	**86.84**	84.64	79.48	**83.62**	81.49

The best results in each column of the table are shown in bold

Table 6 Experimental results of ablation study on CASIE dataset

Model	P (%)	R (%)	F1 (%)
DE3TC	78.97	**80.96**	**79.95**
DE3TC w/o Clue	69.49	70.75	70.11
DE3TC w/o Context	**83.10**	72.21	77.27

The best results in each column of the table are shown in bold

conditions. Clues and contexts contain a wealth of information and work well when the model has a certain amount of training data to help the model correctly identify and classify trigger words.

5 Further Analysis

5.1 Ablation Study

To investigate the impact of each model component, we conduct ablation experiments on ACE 2005 and CASIE for the two main components of the model, the Event Type Information Constructor and Context Selector. When studying the impact of Event Type Information Constructor, we replace the Clue in the event semantic modeling sequence with a query for trigger (**DE3TC w/o Clue**). When studying the impact of Context Selector, we remove the context of event semantics in the event semantic modeling sequence, leaving only Clue and input sentence (**DE3TC w/o Context**). As in the previous section, we also report the performance of the two subtasks, Tri-ID and Tri-CLS, using precision (P), recall (R), and F1 scores (F1) for evaluation. The experimental results are shown in Tables 5 and 6.

From Table 5, we observe that: (1) The DE3TC w/o Clue has a significant decline in Trig-ID (−3.11%) and Trig-CLS (−10.5%) in F1 score. It shows that the Clue plays a crucial

Table 7 F1 scores (%) of Tri-CLS with different proportions of training data

Model	10%	20%	30%
DE3TC	48.90	70.44	73.18
DE3TC w/o Clue	49.52	68.12	71.56
DE3TC w/o Context	**59.83**	**74.15**	**78.28**

The best results in each column of the table are shown in bold

role in both subtasks, especially in trigger classification. The model can learn more accurate semantic information of event type with the Clue. (2) Without descriptions of similar types as context, the performance of DE3TC has a prominent decline in F1 score of the Tri-CLS, indicating that context provides more type-level semantic references for the sentence-level ED, thus helping the model to achieve better performance on trigger classification.

From Table 6, we observe that: (1) On the CASIE dataset, the role of Clue becomes more pronounced. In the absence of Clue, there is a remarkable decrease in F1 score for both subtasks (−9.51% on Tri-ID and −9.84% on Tri-CLS). This indicates the significance of our Clues, especially when dealing with events in certain specific domains. (2) Without descriptions of similar types as context, our method shows a noticeable decrease in performance on the CASIE dataset. This indicates that descriptions of similar types as context can provide substantial assistance on sentence-level ED, including events in specific domains.

5.2 Ablation Under Low-Resource

We also conducted ablation experiments under low-resource conditions on ACE 2005 dataset. The experimental results are shown in Table 7.

From Table 7, it shows that: (1) DE3TC w/o Context performs best, outperforms other baselines on 20% and 30% of training data, and is competitive on 10%. However, the complete DE3TC achieved sub-optimal performance on 20% and 30%, still significantly better than other baselines. An interesting phenomenon is that DE3TC w/o Context performs better than DE3TC. When only a small amount of data is available for training, the Clue captures the event information in the sentence. Under low-resource conditions, the model does not have enough samples to learn the semantic information of event types, and context containing other events' semantics becomes noise. (2) DE3TC w/o Clue performs similarly to DEGREE in Table 4 and is superior to BERT_QA broadly. It shows that our context is of great help when the query does not contain type-specific information under low-resource conditions.

5.3 Research on Hyperparameters

We conduct a series of experiments on the effect of hyperparameters. We extend the length of the keyword list to {1,2,3,4,5} and study its effect on the performance of the model. At the same time, we also studied the impact of the number of event descriptions K in context on the model performance. In the experiment in Chapter 4, we set K to 2. In this section, we set K to 1 as a comparative experiment. Because of the length limit of BERT input, when $K = 3$ or greater, our input will be truncated in most cases, so that context cannot complete the input into BERT.

We show the results in Fig. 3. We observed that: (1) Increasing or decreasing the length of the keyword list negatively impacts the model's performance. When the length is 3, the model achieves the best performance. (2) When $K=1$, the performance of the model decreases

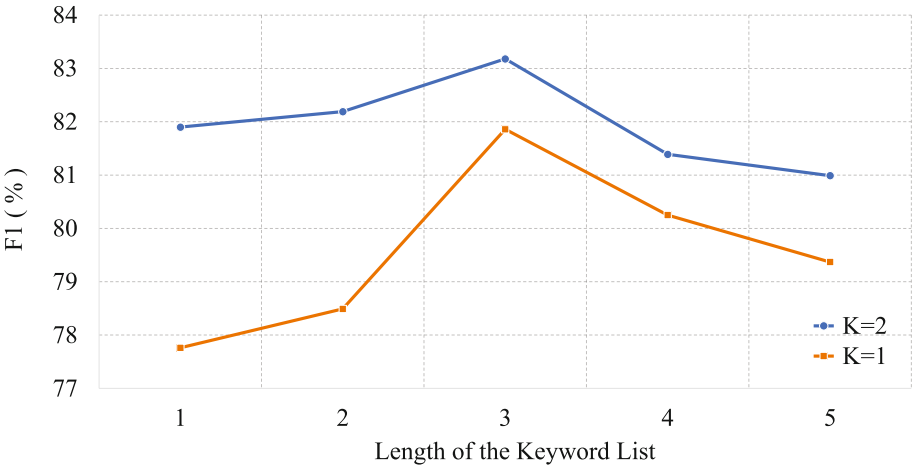


Fig. 3 Performance with different hyperparameters

significantly, with an average decrease of 2.38. This phenomenon indicates that the model cannot fully learn event semantics when the context has only one description of similar event types.

5.4 Research on the Methods of Sentence Encoding

In Sect. 3.3, we use SBERT [30] as our sentence encoder. In this section, we do comparison experiments to investigate the difference in performance between SBERT and the direct method. The direct method for obtaining sentence representations is to use contextual encoder to obtain the representation of all tokens in a sentence, and then use the mean-pooled result of all token representations as mentioned in Sect. 3.3.

When using SBERT, we employed the model with the parameter "roberta-large-nli-stsb-mean-tokens". The meanings of the parameter are as follows: (1) "roberta-large" denotes using RoBERTa-large [8] as the pre-trained model, (2) "nli-stsb" indicates training on Natural Language Inference (NLI) and Semantic Textual Similarity Benchmark (STS-B) task (a task to evaluate the semantic similarity of sentences), and (3) "mean-tokens" indicates the utilization of the mean of representations from all tokens as the representation for the entire sentence.

For the comparison method of sentence encoding, we employed RoBERTa-large as the pre-trained model and utilized the mean of representations from all tokens as the representation for the entire sentence. We use **DE3TC w/o SBERT** to represent the comparison method. Experimental results on ACE 2005 dataset are presented in Table 8.

Table 8 Experimental results with different sentence encoding methods

Model	Tri-ID			Tri-CLS		
	P (%)	R (%)	F1 (%)	P (%)	R (%)	F1 (%)
DE3TC	**85.25**	84.61	**84.93**	**83.5**	**82.87**	**83.18**
DE3TC w/o SBERT	83.74	**85.61**	84.66	81.06	82.67	81.85

The best results in each column of the table are shown in bold

From Table 8, we can observe that SBERT's special network structures and training on the Semantic Textual Similarity task provide significant help in the sentence encoding and the similarity computation. Thereby SBERT provides accurate descriptions of similar events for our method to get the optimal performance.

6 Conclusion and Future Work

In this paper, we propose a novel end-to-end event detection model, DE3TC, capturing event type information by a Clue and accessing the context of event semantics by selecting similar event types.

We propose Clue, which is a sentence capturing interactions between event types and trigger words, thereby containing event type information. Furthermore, we design an Event Type Information Constructor to obtain a simple Clue without the demand of much manual work. In order to access the effective context of event semantics, we design a Context Selector and find similar event types with correlations as context. With type-specific Clue and effective context, the model identifies trigger words and classify them efficiently. The experimental results show that DE3TC has achieved state-of-the-art performance on the ED task. At the same time, DE3TC also performs well under low-resource conditions. In the future, we will study the application of our model in other information extraction tasks and explore the generalization of our approach.

Acknowledgements This research was partially funded by the National Natural Science Foundation of China (NSFC), Nos. 61832014 and 61373165. The authors thank anonymous reviewers for their valuable comments and suggestions.

Author Contributions Conceptualization, LZ and GR; methodology, BL and GR; supervision, XW and GR; writing-original draft, BL, and LZ; writing-review and editing, GR and QC. All authors have read and agreed to the published version of the manuscript.

Declarations

Conflict of interest The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Saurí R, Knippen R, Verhagen M et al (2005) Evita: a robust event recognizer for QA systems. In: HLT/EMNLP 2005, human language technology conference and conference on empirical methods in natural language processing, proceedings of the conference, 6–8 October 2005, Vancouver, British Columbia, Canada. The Association for Computational Linguistics, pp 700–707. https://aclanthology.org/H05-1088/
2. Wan L, Liao J, Zhu X (2009) A frequent pattern based framework for event detection in sensor network stream data. In: Omitaomu OA, Ganguly AR, Gama J et al (eds) Proceedings of the third international

workshop on knowledge discovery from sensor data, Paris, France, June 28, 2009. ACM, pp 87–96. https://doi.org/10.1145/1601966.1601982
3. Qin Y, Zhang Y, Zhang M et al (2013) Feature-rich segment-based news event detection on twitter. In: Sixth international joint conference on natural language processing, IJCNLP 2013, Nagoya, Japan, October 14–18, 2013. Asian Federation of Natural Language Processing/ACL, pp 302–310. https://aclanthology.org/I13-1035/
4. Chen Y, Xu L, Liu K et al (2015) Event extraction via dynamic multi-pooling convolutional neural networks. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing of the Asian Federation of natural language processing, ACL 2015, July 26–31, 2015, Beijing, China, Volume 1: long papers. The Association for Computer Linguistics, pp 167–176. https://doi.org/10.3115/V1/P15-1017
5. Nguyen TH, Cho K, Grishman R (2016) Joint event extraction via recurrent neural networks. In: Knight K, Nenkova A, Rambow O (eds) NAACL HLT 2016, The 2016 conference of the North American chapter of the association for computational linguistics: human language technologies, San Diego California, USA, June 12–17, 2016. The Association for Computational Linguistics, pp 300–309. https://doi.org/10.18653/V1/N16-1034
6. Cui S, Yu B, Liu T et al (2020) Edge-enhanced graph convolution networks for event detection with syntactic relation. In: Cohn T, He Y, Liu Y (eds) Findings of the association for computational linguistics: EMNLP 2020, Online Event, 16–20 November 2020, findings of ACL, vol EMNLP 2020. Association for Computational Linguistics, pp 2329–2339. https://doi.org/10.18653/V1/2020.FINDINGS-EMNLP.211
7. Devlin J, Chang M, Lee K et al (2019) BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein J, Doran C, Solorio T (eds) Proceedings of the 2019 conference of the North American Chapter of the association for computational linguistics: human language technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2–7, 2019, Volume 1 (long and short papers). Association for Computational Linguistics, pp 4171–4186. https://doi.org/10.18653/V1/N19-1423
8. Liu Y, Ott M, Goyal N et al (2019) Roberta: a robustly optimized BERT pretraining approach. arXiv:1907.11692
9. Lan Z, Chen M, Goodman S et al (2020) ALBERT: a lite BERT for self-supervised learning of language representations. In: 8th international conference on learning representations, ICLR 2020, Addis Ababa, Ethiopia, April 26–30, 2020. OpenReview.net. https://openreview.net/forum?id=H1eA7AEtvS
10. Lewis M, Liu Y, Goyal N et al (2020) BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: Jurafsky D, Chai J, Schluter N et al (eds) Proceedings of the 58th annual meeting of the association for computational linguistics, ACL 2020, Online, July 5–10, 2020. Association for Computational Linguistics, pp 7871–7880. https://doi.org/10.18653/V1/2020.ACL-MAIN.703
11. Vaswani A, Shazeer N, Parmar N et al (2017) Attention is all you need. In: Guyon I, von Luxburg U, Bengio S et al (eds) Advances in neural information processing systems 30: annual conference on neural information processing systems 2017, December 4–9, 2017, Long Beach, CA, USA, pp 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html
12. Lin Y, Ji H, Huang F et al (2020) A joint neural model for information extraction with global features. In: Jurafsky D, Chai J, Schluter N et al (eds) Proceedings of the 58th annual meeting of the association for computational linguistics, ACL 2020, Online, July 5–10, 2020. Association for Computational Linguistics, pp 7999–8009. https://doi.org/10.18653/V1/2020.ACL-MAIN.713
13. Wadden D, Wennberg U, Luan Y et al (2019) Entity, relation, and event extraction with contextualized span representations. In: Inui K, Jiang J, Ng V et al (eds) Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3–7, 2019. Association for Computational Linguistics, pp 5783–5788. https://doi.org/10.18653/V1/D19-1585
14. Du X, Cardie C (2020) Event extraction by answering (almost) natural questions. In: Webber B, Cohn T, He Y et al (eds) Proceedings of the 2020 conference on empirical methods in natural language processing, EMNLP 2020, Online, November 16–20, 2020. Association for Computational Linguistics, pp 671–683. https://doi.org/10.18653/V1/2020.EMNLP-MAIN.49
15. Liu J, Chen Y, Liu K et al (2020) Event extraction as machine reading comprehension. In: Webber B, Cohn T, He Y et al (eds) Proceedings of the 2020 conference on empirical methods in natural language processing, EMNLP 2020, Online, November 16–20, 2020. Association for Computational Linguistics, pp 1641–1651. https://doi.org/10.18653/V1/2020.EMNLP-MAIN.128
16. Li F, Peng W, Chen Y et al (2020) Event extraction as multi-turn question answering. In: Cohn T, He Y, Liu Y (eds) Findings of the association for computational linguistics: EMNLP 2020, Online Event,

16–20 November 2020, Findings of ACL, vol EMNLP 2020. Association for Computational Linguistics, pp 829–838. https://doi.org/10.18653/V1/2020.FINDINGS-EMNLP.73
17. Li S, Ji H, Han J (2021) Document-level event argument extraction by conditional generation. In: Toutanova K, Rumshisky A, Zettlemoyer L et al (eds) Proceedings of the 2021 conference of the North American Chapter of the association for computational linguistics: human language technologies, NAACL-HLT 2021, Online, June 6–11, 2021. Association for Computational Linguistics, pp 894–908. https://doi.org/10.18653/V1/2021.NAACL-MAIN.69
18. Lu Y, Lin H, Xu J et al (2021) Text2event: controllable sequence-to-structure generation for end-to-end event extraction. In: Zong C, Xia F, Li W et al (eds) Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing, ACL/IJCNLP 2021, (Volume 1: long papers), Virtual Event, August 1–6, 2021. Association for Computational Linguistics, pp 2795–2806. https://doi.org/10.18653/V1/2021.ACL-LONG.217
19. Hsu I, Huang K, Boschee E et al (2022) DEGREE: a data-efficient generation-based event extraction model. In: Carpuat M, de Marneffe M, Ruíz IVM (eds) Proceedings of the 2022 conference of the North American Chapter of the association for computational linguistics: human language technologies, NAACL 2022, Seattle, WA, United States, July 10–15, 2022. Association for Computational Linguistics, pp 1890–1908. https://doi.org/10.18653/V1/2022.NAACL-MAIN.138
20. Liu P, Yuan W, Fu J et al (2023) Pre-train, prompt, and predict: a systematic survey of prompting methods in natural language processing. ACM Comput Surv 55(9):195:1-195:35. https://doi.org/10.1145/3560815
21. Du X, Li S, Ji H (2022) Dynamic global memory for document-level argument extraction. In: Muresan S, Nakov P, Villavicencio A (eds) Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers), ACL 2022, Dublin, Ireland, May 22–27, 2022. Association for Computational Linguistics, pp 5264–5275. https://doi.org/10.18653/V1/2022.ACL-LONG.361
22. Pour SN, Hosseini S, Hua W et al (2022) Soulmate: short-text author linking through multi-aspect temporal-textual embedding. IEEE Trans Knowl Data Eng 34(1):448–461. https://doi.org/10.1109/TKDE.2020.2982148
23. Liu Y, Hua W, Xin K et al (2023) TEA: time-aware entity alignment in knowledge graphs. In: Ding Y, Tang J, Sequeda JF et al (eds) Proceedings of the ACM web conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023–4 May 2023. ACM, pp 2591–2599. https://doi.org/10.1145/3543507.3583317
24. Feng K, Rao G, Zhang L et al (2023) An interlayer feature fusion-based heterogeneous graph neural network. Appl Intell 53(21):25,626-25,639. https://doi.org/10.1007/S10489-023-04840-W
25. Lyu P, Rao G, Zhang L et al (2023) Bilgat: bidirectional lattice graph attention network for Chinese short text classification. Appl Intell 53(19):22,405-22,414. https://doi.org/10.1007/S10489-023-04700-7
26. Saaki M, Hosseini S, Rahmani S et al (2023) Value-wise convnet for transformer models: an infinite time-aware recommender system. IEEE Trans Knowl Data Eng 35(10):9932–9945. https://doi.org/10.1109/TKDE.2022.3219231
27. Hosseini S, Yin H, Zhou X et al (2019) Leveraging multi-aspect time-related influence in location recommendation. World Wide Web 22(3):1001–1028. https://doi.org/10.1007/S11280-018-0573-2
28. Liu X, Huang H, Shi G et al (2022) Dynamic prefix-tuning for generative template-based event extraction. In: Muresan S, Nakov P, Villavicencio A (eds) Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers), ACL 2022, Dublin, Ireland, May 22–27, 2022. Association for Computational Linguistics, pp 5216–5228. https://doi.org/10.18653/V1/2022.ACL-LONG.358
29. Sheng J, Sun R, Guo S et al (2022) CorED: incorporating type-level and instance-level correlations for fine-grained event detection. In: Amigó E, Castells P, Gonzalo J et al (eds) SIGIR '22: the 45th international ACM SIGIR conference on research and development in information retrieval, Madrid, Spain, July 11–15, 2022. ACM, pp 1122–1132. https://doi.org/10.1145/3477495.3531956
30. Reimers N, Gurevych I (2019) Sentence-bert: sentence embeddings using siamese bert-networks. In: Inui K, Jiang J, Ng V et al (eds) Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3–7, 2019. Association for Computational Linguistics, pp 3980–3990. https://doi.org/10.18653/V1/D19-1410
31. Satyapanich T, Ferraro F, Finin T (2020) CASIE: extracting cybersecurity event information from text. In: The thirty-fourth AAAI conference on artificial intelligence, AAAI 2020, the thirty-second innovative applications of artificial intelligence conference, IAAI 2020, the tenth AAAI symposium on educational advances in artificial intelligence, EAAI 2020, New York, NY, USA, February 7–12, 2020. AAAI Press, pp 8749–8757. https://doi.org/10.1609/AAAI.V34I05.6401
32. Lu Y, Liu Q, Dai D et al (2022) Unified structure generation for universal information extraction. In: Muresan S, Nakov P, Villavicencio A (eds) Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers), ACL 2022, Dublin, Ireland, May 22–27, 2022. Asso-

ciation for Computational Linguistics, pp 5755–5772. https://doi.org/10.18653/V1/2022.ACL-LONG.395

33. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In: Bengio Y, LeCun Y (eds) 3rd international conference on learning representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, conference track proceedings. arXiv:1412.6980
34. Paolini G, Athiwaratkun B, Krone J et al (2021) Structured prediction as translation between augmented natural languages. In: 9th international conference on learning representations, ICLR 2021, Virtual Event, Austria, May 3–7, 2021. OpenReview.net. https://openreview.net/forum?id=US-TP-xnXI
35. Li R, Zhao W, Yang C et al (2021) Treasures outside contexts: improving event detection via global statistics. In: Moens M, Huang X, Specia L et al (eds) Proceedings of the 2021 conference on empirical methods in natural language processing, EMNLP 2021, virtual event/Punta Cana, Dominican Republic, 7–11 November, 2021. Association for Computational Linguistics, pp 2625–2635. https://doi.org/10.18653/V1/2021.EMNLP-MAIN.206

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.