applied sciences

MDPI

Article

Type Hierarchy Enhanced Event Detection without Triggers

Youcheng Yan [1,2,3,†], Zhao Liu [1,2,3,†], Feng Gao [1,2,3] and Jinguang Gu [1,2,3,*]

[1] College of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430065, China
[2] Institute of Big Data Science and Engineering, Wuhan University of Science and Technology, Wuhan 430065, China
[3] Key Laboratory of Rich-Media Knowledge Organization and Service of Digital Publishing Content, National Press and Publication Administration, Beijing 100038, China
* Correspondence: simon@ontoweb.wust.edu.cn
† These authors contributed equally to this work.

Abstract: Event detection (ED) aims to detect events from a given text and categorize them into event types. Most of the current approaches to ED rely heavily on the human annotations of triggers, which are often costly and affect the application of ED in other fields. However, triggers are not necessary for the event detection task. We propose a novel framework called Type Hierarchy Enhanced Event Detection Without Triggers (THEED) to avoid this problem. More specifically, We construct a type hierarchy concept module using the external knowledge graph Probase to enhance the semantic representation of event types. In addition, we divide input instances into word-level and context-level representations, which can make the model use different level features. The experimental result indicates that our proposed approach achieves better improvement. Additionally, it is significantly competitive with mainstream trigger-based models.

Keywords: hierarchy concept; attention mechanism; probase; bi-LSTM; event detection

Citation: Yan, Y.; Liu, Z.; Gao, F.; Gu, J. Type Hierarchy Enhanced Event Detection without Triggers. *Appl. Sci.* **2023**, *13*, 2296. https://doi.org/10.3390/app13042296

Academic Editors: Gui-Lin Qi, Tong Xu and Meng Wang

Received: 8 January 2023
Revised: 5 February 2023
Accepted: 8 February 2023
Published: 10 February 2023

1. Introduction

Event extraction is the task of extracting structure information of events from unstructured text [1–3], which has a wide range of usages in fields such as financial analysis [4], fake news detection [5] and social emergency event judgment [6]. It aims to detect events with triggers and their corresponding arguments. Event detection, as a core subtask of event extraction, has attracted many researchers' attention in recent years. Since most existing studies consider triggers as essential features of the event type, the event detection task is modeled as predicting whether each word in a given sentence is a trigger and what type of event it triggers [7–9]. Consequently, these methods often require the annotation of triggers for each sentence in the training set.

However, triggers are not necessary for event detection tasks. Firstly, It is difficult to annotate all the triggers in some sentences, especially long ones. As shown in Figure 1, there are three events in S1, but a trigger such as "detonated" may be omitted in the annotation process. Secondly, as the statistical results in Figure 2a,b show, some triggers can easily correspond to event types, such as "married", but there are still some triggers with ambiguous meanings, so it is difficult to correspond to event types. For example, "went" in Figure 1 S3 is difficult to correspond to the event type "Arrest-Jail." Finally, recall that event detection aims to identify events and classify them into event types; identifying triggers is only an intermediate process [10].

To avoid the large amount of resources consumed in annotating triggers, errors in the incorrect annotation of triggers and word ambiguity in the subsequent event detection task, our proposed approach will complete the event detection task without trigger annotations. However, studies on this task are still in the initial stage [11,12]. All the event types will be transformed into low-dimensional vectors by looking up a randomly initialized embedding

table. Still, the semantic information of event types represented in this way is limited. For each event type, we can find its corresponding superordinate and subordinate concepts to enrich its semantic information, so this paper proposes a type hierarchy concept module to enhance the semantic representation of event types and compensate for some semantic features lost by event triggers. Specifically, it uses the IS_A relationship in the probabilistic knowledge graph Probase [13] to look up the hierarchy concepts of an event type while calculating their confidence based on the frequency of occurrence [14]. Unlike the remote supervision framework [15], our approach does not utilize an external knowledge base to annotate text or generate a new training corpus for event detection.

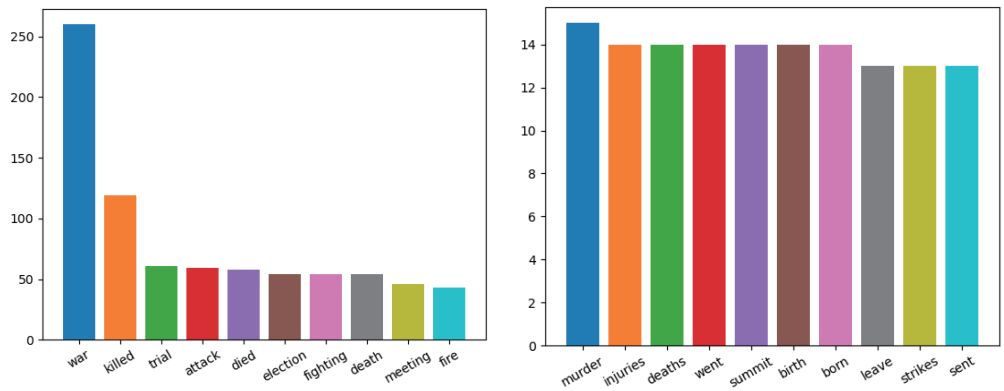
Another limitation is that existing studies consider only context-level features and ignore word-level features in the event detection task [16,17] or consider every word as a potential trigger when using word-level features [9,18]. The former will lose an essential part of the features during the event detection task, while the latter will take into account many useless words such as "a", "the", etc.

S1: Three people were killed and about 50 injured when another Briton, Asif Muhammad Hanif, 21, from London, detonated explosives strapped to his body.
Event-Type: Die; Attack; Injure

S2: They got married in 1985.
Event-Type: Marry

S3: The company's performance wasn't all that hot even before the CEO went to prison...
Event-Type: Arrest-Jail

Figure 1. Event instances from the ACE 2005 benchmark, where the trigger words are colored.



(**a**) (**b**)

Figure 2. Statistics about the number of occurrences of triggers in the ACE2005 dataset. (**a**) Top 10 occurrences of triggers. (**b**) Top 50–60 occurrences of triggers.

To this end, we propose a new event detection framework with enhanced type hierarchy called THEED. Our framework represents event types as hierarchy concepts with rich semantics. The hierarchy concepts are then used to complete event detection by leveraging the attention mechanism to capture their interactions with word-level and context-level representations. Specifically, (1) for the type hierarchy concept module, we construct a type hierarchy network by looking for the superordinate and subordinate concepts of each event type using the probabilistic knowledge graph Probase. Furthermore, we calculate

the confidence of each hierarchy concept to indicate the similarity with the event type. (2) For word-level and context-level representations, our framework first converts the input tokens into embeddings and then applies the bi-LSTM layer to calculate the context-level representation. As for word-level representation, we exploit the attention mechanism to obtain top K contribution to the sentence's semantic information. (3) For event detection, we give an input sentence, and we make the binary decoding of each representation based on the hierarchy concept. Our contributions can be summarized as follows:

- We propose a trigger-free event detection framework with the semantic enhancement of event types called THEED.
- We design a type hierarchy concept enhanced module that uses the Probase probabilistic knowledge graph to construct hierarchical concepts with confidence. It enhances the semantic representation of event types and improves the accuracy of the event detection task.
- We exploit the attention mechanism to divide the input into word-level and context-level representations, which consider different levels of features and filter useless words.

The rest of the paper is organized as follows. First of all, we apply the related work in event detection with triggers and event detection without triggers in Section 2. Secondly, in Section 3, we introduce our proposed framework in detail. Again, in Section 4, we present the experimental details and evaluate the results. Then, Section 3 gives a concrete example and visualizes the results. In Section 4, we analyze errors and give possible reasons according to the event detection results. Finally, Section 5 gives the conclusion of this framework and considers its prospects.

2. Related Work

The main purpose of this section is to review the related work on event detection. It is mainly divided into two subsections: event detection with triggers and event detection without triggers. We also introduce the semantically enhanced ED methods in Section 2.1.

2.1. Event Detection with Triggers

Event detection has been used as a fundamental core technology in NLP. Early approaches are feature-based. Researchers construct rules manually or automatically and then use them as templates to guide event detection [19–22]. However, rules and features greatly affect the overall performance of the ED model and require a high level of human resources and expertise. With the gradual improvement of deep learning theory and technology in recent years, more and more approaches use deep learning methods to complete ED. Convolutional neural networks capture semantic associations between consecutive words by convolutional operations. Chen et al. [8] proposed a dynamic multipooling convolutional neural network, which uses a dynamic multipooling layer based on event triggers and arguments, to keep more crucial information. Nguyen et al. [23] used a joint framework with bidirectional recurrent neural networks to perform event extraction. The framework can learn hidden feature representations automatically from data according to the continuous and generalized representations of words. Feng et al. [24] constructed a hybrid neural network to capture both sequence and chunk information from specific contexts.

On the other hand, there are some approaches used to enhance semantic features during event detection. A hierarchical and supervised attention model is proposed by Zhao et al. [25], which pays word-level attention to triggers and sentence-level attention to sentences containing events. Moreover, the use of ontology in various fields is increasing [26–29], and some studies obtain satisfactory results with ontology. For example, Deng et al. [29] formulate event detection as a process of event ontology population, which associates an event instance with a predefined event type. Most recently, Wang et al. [9] proposed a framework that uses event types as queries to extract candidate trigger words. Nevertheless, this framework formulates the query based on its event type name and the top K of prototype triggers.

A common problem with the above approaches is that they rely too heavily on the annotated triggers. However, annotating is usually resource-intensive and may cause error propagation if incorrectly annotated. Unlike previous work, our framework accomplishes event detection without triggers.

2.2. Event Detection without Triggers

To reduce the massive annotation of triggers in new domains, several studies have been exploring event detection without triggers. Xu et al. [30] constructed a deep learning model called BLMA, which uses bi-LSTM and multihead weighted attention to perform event detection on an automatically tagged screenplay dataset. Liu et al. [10] propose the TBNNAM framework, which encodes a sentence based on event types via the attention mechanism on an LSTM model. This framework does not utilize bi-LSTM, which makes it capture limited contextual information. Moreover, individual event types represent fewer features, which leads to worse performance than trigger-based methods. Hsu et al. [31] feed inputs and a manually predefined template into a generation-based neural network to output a natural sentence for further event extraction. Nevertheless, The template is event-type-specific, manually created and the process is more complex. Lu et al. [32] propose a sequence-to-structure model to avoid using labeled triggers to extract events in an end-to-end manner. Although the model does not use token-level annotations such as trigger offset, the framework still requires much effort to identify trigger words.

In general, there are few methods for event detection without triggers. The above approaches either consider only context-level features or consider every word as a potential trigger when using word-level features. On the other hand, most embedding models typically represent each event type only using the literal meaning of the words or random initialization, which makes these models represent a limited semantic representation of event types.

3. Methodology

3.1. Task Definition

In this section, we briefly introduce the task definition to facilitate understanding the following subsections. Following the previous work [10,19], we are given a training dataset, $\{(x_i, y_i)\}_{i=1}^{N}$, wherein N denotes the number of sentence–event pairs. x_i is the i-th sentence. Let $x_i = w_1, w_2, \cdots, w_L$ be a sentence of L words, and the y_i is the corresponding event type, where $y_i \in Y_s$, which indicates an event label sequence $[y_1, y_2, \cdots, y_N]$. We treat Y_s simply as 34 distinct event types, including 33 event types based on the ACE 2005 corpus and an "NA" to denote that sentences do not contain any events. Our goal is to train a model to detect the corresponding event type, which may contain one or more events, as accurately as possible given the input data. It should be noted that the training set does not contain annotated triggers. Consider the following sentence, "He claimed Iraqi troops had destroyed five tanks", our eventual goal is to detect the "Attack" event.

Our task can be divided into (1) how to capture the effective features in the input text, (2) how to learn more precise representations to embed semantic information between input sentences and event types and (3) how to detect each event in a sentence which may contain an arbitrary number of events.

3.2. Model Overview

In this section, we introduce the framework of THEED in detail. Without event triggers, there will be three challenges, as shown in the task definition. To solve the first challenge, we divided the features into context-level and word-level, where context-level captures contextual information and word-level denotes word information. Meanwhile, we remove useless words from the sentence, such as "a", "the", etc., using the attention mechanism. As for the second challenge, rather than a simple "event-type embedding table", we first construct the type graphs, and then assign confidence to each hierarchical concept to strengthen the semantic relationship between the input text and the event type.

Finally, to solve the multievent problem, we convert multilabel classification into a multiple binary classification task. For example, consider the following sentence, "We go to war in Iraq, 200,000 people start protesting in Pakistan, they put too much pressure on the government", a perfect model should detect two different events from the sentence: an Attack event and a Demonstrate event.

As Figure 3 shows, THEED comprises three components: (1) The instance encoding represents a sentence into hidden embeddings. (2) The type hierarchy concept modular component enhances the semantic representation of event types through an external knowledge graph. (3) The event detection task relies on instance encoding and the hierarchy concept embedding to estimate the probability of a certain event type for the sentence. The details are as follows.

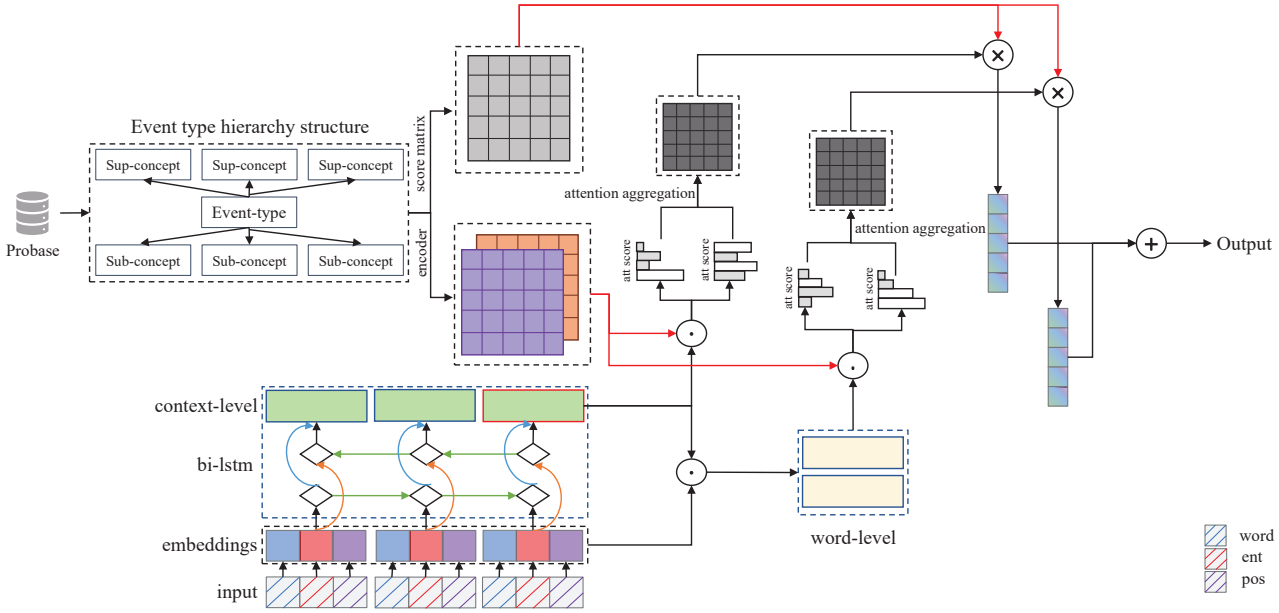


Figure 3. The architecture overview of THEED. It consists of three components. Component 1: Instance encoding. Component 2: Type hierarchy concept module. Component 3: Binary encoding. (The type hierarchy concept module only indicates one example. Due to space constraints, confidence is not marked on the lines.)

3.3. Instance Encoding

Given a sentence $W = w_1, w_2, \cdots, w_n$, the ACE2005 corpus is not only annotated with events but also with entities. Following previous approaches, we will use both entity tags and part-of-speech tags in our framework [8–10]. As for pos tags, we exploit StanfordCoreNLP (https://stanfordnlp.github.io/CoreNLP/ (accessed on 1 January 2023)) to recognize them.

3.3.1. Word/Entity/Pos Embeddings

Word embeddings have been shown to be able to capture the complex semantic regularities of words from massive unlabeled data [33]. In our work, we use the Skip-gram [34] (accessed on 1 January 2023) model to learn word embeddings in New York Times Corpus(NYT) https://catalog.ldc.upenn.edu/LDC2008T19. The reason why we chose NYT is that it is large and diverse; moreover, its data domain is similar than the ACE2005 dataset, which is helpful [35]. As for entity tags and pos tags, we randomly initialize embedding tables for them. Then, every word token and the tag will convert to a low-dimensional dense vector. Finally, the input token embedding will consist of three parts: (1) word embeddings, where the dimension is dw; (2) entity embeddings, where the dimension is de; and (3) pos embeddings, where the dimension is dp.

3.3.2. Context-Level Representation

Long short-term memory (LSTM) [36] is the most widely used method to deal with long-term dependence. It consists of three gates (i.e., an input gate, a forget gate and an output gate) and a storage unit to hold the state of each neuron. The standard LSTM can only utilize the historical information of the sequences, which are single-directional features. Due to the complexity of natural language sentence structure, it is also necessary to consider future information.

As Figure 3 shows, we exploit bi-LSTM [37], which uses two LSTMs to work in the forward and backward time directions to obtain contextual information. First, we feed sequence embedding x to forward propagation, which runs x from the past to the future. Similarly, we feed the reverse order to backward propagation, which considers the contextual information from the future to the past. Therefore, we can preserve information in two directions at any point in time.

Given the input token w, the input of bi-LSTM is x_t, which includes three parts (i.e., a word embedding e_{wt}, an entity embedding e_{et}, and a pos tag embedding e_{pt}), and its hidden state is h_t at time t. The formula is expressed as follows:

$$x_t = concat(e_{wt}; e_{et}; e_{pt}) \tag{1}$$

$$\overrightarrow{h_t} = LSTM(x_t, \overrightarrow{h_{t-1}}) \tag{2}$$

$$\overleftarrow{h_t} = LSTM(x_t, \overleftarrow{h_{t+1}}) \tag{3}$$

$$h_t = [\overrightarrow{h_t} \oplus \overleftarrow{h_t}] \tag{4}$$

Here, we obtain the hidden embeddings $h = \{h_1, h_2 \cdots h_i \cdots h_n\}$ of the input tokens, where we use the last hidden state h_n to denote the context-level representation, because it contains all the information in the forward and backward directions.

3.3.3. Word-Level Representation

Existing studies generally only adopt context-level representation and ignore word-level, or each word is considered at the word level. However, it will contain useless words such as "the',' "an", "a", etc. We introduce attention mechanisms in our framework to select words with top K (in our experiments, we set K = 8) contribution to learning a weight distribution. In this way, we can filter out the useless words in the sentence. We give an example and visualize it later in the case study section.

$$\alpha^j = \frac{exp(h_j \cdot h_{context}^T)}{\sum_{i=1}^{n} exp(h_i \cdot h_{context}^T)} \tag{5}$$

$$h_{word} = gather(h, topK(a^j).indices) \tag{6}$$

where the attention vector α is calculated based on hidden states h and the context representation $h_{context}$. Specifically, the attention score calculation of the j-th input token is illustrated in Equation (5). Then, we use the top K operation to obtain the K indices of hidden states by contribution ranking. Finally, we use the gather operation to obtain the hidden state corresponding to the indices. It is noted that both top K and gather are functions that come with the framework codebase. we obtain the word-level representation $h_{word} = \{h_1, h_2 \cdots h_k\}$.

3.4. Type Hierarchy Concept Module

In order to learn more precise semantic representations between input sentences and event types, we construct a hierarchical network for each event type and calculate the confidence of each hierarchical concept to indicate the similarity to the event type. Finally, we use a scaled dot-product attention mechanism to obtain a single hierarchical concept embedding from the token embedding sequence. The type hierarchy concept module consists of three parts: concept graphs, concept confidence and concept representation.

3.4.1. Hierarchy Concept Graphs

As several studies have shown, a simple way to represent an event type is to use its event type name directly [10,38]. However, this approach means limited semantics information, and the names often contain ambiguities, which may introduce errors in the later event detection task. Inspired by previous methods [9,39], we use type hierarchy concept graphs to enrich the semantics of each event type.

Figure 3 illustrates that we exploit the external knowledge graph Probase to build hierarchy concept graphs. Probase is a large-scale concept knowledge graph proposed by Microsoft, containing 5,401,933 concepts, 12,551,613 instances and 87,603,947 IsA triples [40]. Figure 4 shows partial examples in the Probase knowledge graph. By using this KG, concepts can be mapped to different semantic concepts. Each event type, such as Arrest-Jail, is represented with a text, including a shortlist of superordinate concepts and a shortlist of subordinate concepts, such as "legal problem" and "rape suspect", which select from the Probase through IsA relation. The reason why we use superordinate concepts and subordinate concepts is that they can extend the concept of event types or instantiate the concept of event types. Finally, we use $sup_t = \{sup_{t1}, sup_{t2} \cdots sup_{tn}\}$ (In our experiments, we set $n = 20$) to denote the set of superordinate concepts of event type t, and similarly, use $sub_t = \{sub_{t1}, sub_{t2} \cdots sub_{tn}\}$ to indicate the set of subordinate concepts.

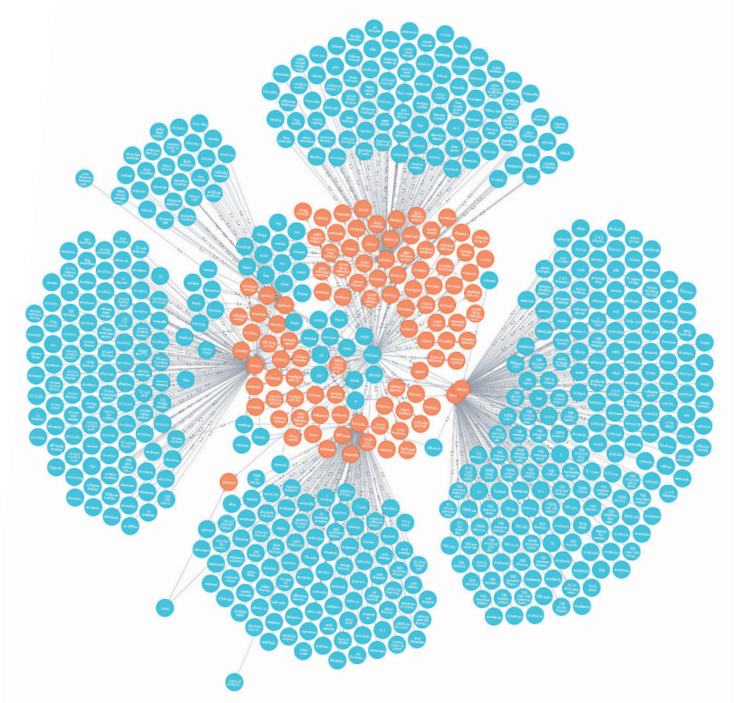


Figure 4. Partial display of the Probase KG (concepts in orange and instances in blue) connected by the "IS_A" relationship.

3.4.2. Hierarchy Concept Confidence

The hierarchy concepts should not only indicate whether the sup_t or sub_t is related to an event type but also calculate the confidence of how likely sup_t/sub_t could be mapped to

type in the Probase. We formulate the hierarchy concepts confidence of sup_t/sub_t as follows:

$$superordinate(sup_{ti}, evt_t) = \frac{freq(sup_{ti}, evt_t)}{\sum_{k=1} freq(sup_{ti}, evt_k)} \tag{7}$$

$$subordinate(sub_{ti}, evt_t) = \frac{freq(sub_{ti}, evt_t)}{\sum_{k=1} freq(sub_{ti}, evt_k)} \tag{8}$$

Wherein $freq(sup_{ti}, evt_t)$ is derived from the Probase, and it indicts the frequency between sup_{ti}/sub_{ti} and evt_t. As shown in the previous sections, the hierarchy concepts of an event type will be a list, so their confidence is also a list, denoted by $SupScore_t$ and $SubScore_t$. Table 1 shows that we construct the superordinate and subordinate concepts of the event type "Arrest-Jail" and assign confidence to the hierarchical concepts. Due to space constraints, we only display part of the hierarchical concepts.

Table 1. Hierarchical concept and confidence for an example of "Arrest-Jail"; due to space constraints, we only display part of the hierarchical concept.

Event Type	Superordinate Concept	Confidence	Subordinate Concept	Confidence
Arrest-Jail	legal problem	0.15157	los angeles county jail	0.09714
	terminal event	0.13662	rape suspect	0.08928
	punishment	0.07898	county jail	0.07142
	...	...	...	...
	public facility	0.07590	east cambridge jail	0.05357

3.4.3. Hierarchy Concept Representation

We can see from Table 1 that the hierarchical concepts may contain a different number of words. So, it is important to obtain a single embedding from a sequence of embeddings with the same embedding size. Specifically, given a superordinate concept or subordinate concept with a sequence of embeddings, $x = x_1, x_2, x_3, \cdots, x_{N_x}$, where the dimension of each x_i is d_w, and x is an embedding with size $d_w \times N_x$, then we use the dot-product attention [41] to produce a single embedding from a sequence of embeddings.

$$hie_i = \frac{Q^{\mathrm{T}} x_i}{\sqrt{d_w}} \tag{9}$$

$$\alpha_i = \frac{\exp(hie_i)}{\sum_j \exp(hie_i)} \tag{10}$$

$$\overline{x} = Dropout(LayerNorm(\sum_i \alpha_i x_i)) \tag{11}$$

Wherein the dimension of the single embedding $\overline{x}$ is d_w, and Q is a trainable parameter. Dividing by $\sqrt{d}$ is to alleviate the problem of gradient disappearance if the dot product is large. In addition, we add layer normalization and dropout to avoid overfitting. In our experiments, we also exploit max pooling or mean pooling to achieve the same result. According to experimental results, we find that this method performed similarly to max pooling, while sometimes slightly higher than the max pooling, and both of them performed better than the mean pooling. To make the model work better, we use this method to obtain a single embedding from a sequence of embeddings.

3.5. Event Detection

Given an input sentence, we aim to detect the corresponding event type automatically. In order to achieve this goal, we should capture the semantic correlation of instance encoding to the event type. Thus, we adopt an attention mechanism to learn a weight distribution over the two-level representation of the event type and obtain an event type

aware two-level representation, where the two-level representation refers to context-level representation and word-level representation.

$$A_{word} = \alpha^{\mathrm{T}} \cdot h_{word} \tag{12}$$

$$\alpha^k = \frac{\exp(h_k \cdot t^{\mathrm{T}})}{\sum_i^K \exp(h_i \cdot t^{\mathrm{T}})} \tag{13}$$

$$A_{context} = h_{context} \cdot t^{\mathrm{T}} \tag{14}$$

where $\alpha = [\alpha_1, \alpha_2 \cdots \alpha_k]$ is the attention vector, and A_{word} denotes the event type aware word-level representation. Similarly, we use $A_{context}$ to indicate the event type aware context-level representation. The semantic relevance captured using only the event type provided by ACE205 is limited, so we use the type-hierarchy-enhanced concepts to complete event detection without triggers after obtaining the word-level and context-level type aware representations.

$$s_{word}^{sup} = \sigma(A_{word} W V_{sup}) \tag{15}$$

$$y_{word}^{sup} = \sum s_{word}^{sup} SupScore \tag{16}$$

Wherein W is a learnable parameter matrix, V_{sup} is a superordinate concept representation matrix and σ is the Sigmoid function. y_{word}^{sup} is one of the outputs, which is designed to capture the feature between word-level representation and the superordinate concept. Similarly, due to the space limitations, we use y_{word}^{sub} , $y_{context}^{sup}$ and $y_{context}^{sub}$ to denote the results of the other three levels. Finally, y is defined as the weighted sum of four outputs.

$$y = \lambda(y_{word}^{sup} + y_{word}^{sub}) + (1 - \lambda)(y_{context}^{sup} + y_{context}^{sub}) \tag{17}$$

Here, $\lambda \in [0, 1]$ is a hyperparameter for trade-off between word-level outputs and context-level outputs. To determine whether the input sentence contains a certain event type, we follow the settings in [10] as the following equation, where the number 1 indicates that the sentence includes an event and otherwise denotes the label of "NA".

$$\widetilde{y} = \begin{cases} 0 & y < 0.5 \\ 1 & \text{otherwise} \end{cases} \tag{18}$$

In order to solve the multilabel problem in the input instance, our framework converts multilabel classification into binary decoding. As shown in Table 2, each training sample is a <sentence instance, event type> pair. Assume that the sentence has three predefined event types $t1$, $t2$, $t3$, and $t1$, $t3$ are correct. Then, the label is 1, 0, 1.

Table 2. Example of binary classification of sentences.

Training Sample	Label
<s, $t1$>	1
<s, $t2$>	0
<s, $t3$>	1

After analyzing the training corpus, we find that the quantity of positive samples is much less than negative samples, and in general, positive samples will contain more information. So, we use a bias term to strengthen the positive samples [42]. The training objective is to minimize the following loss function:

$$\mathcal{L} = \frac{1}{T} \sum_{i=1}^{T} (y(x^i) - \widehat{y}^i)^2 (1 + \widehat{y}^i \cdot \beta) + \delta ||\theta||^2 \tag{19}$$

Here, x is the training instances, $\widehat{y} \in \{0, 1\}$ is the label tag and $(1 + \widehat{y}^i \cdot \beta)$ is the bias term. θ is the parameter and $\delta > 0$ is the weight of the L2 normalization term, which we exploit to prevent overfitting.

4. Experiments and Evaluation

4.1. Dataset and Baselines

In this paper, we conduct extensive experiments on the ACE2005 dataset [43], which contains 599 English documents and defines 33 event types. We split the dataset following the previous event extraction work [1,8,22]. To evaluate our model, we adopt precision (P), recall (R) and F1 scores, among which F1 is the most comprehensive measurement.

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \times 100\% \tag{20}$$

$$\mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \times 100\% \tag{21}$$

$$\mathrm{F1} = \frac{2 \times \mathrm{P} \times \mathrm{R}}{\mathrm{P} + \mathrm{R}} \times 100\% \tag{22}$$

where TP, FP and FN denote true positives, false positives and false negatives, respectively.

It is important to emphasize that our framework finishes event detection without triggers. Therefore, we remove trigger annotations from the corpus. Finally, each sentence is assigned only a set of labels based on event type annotations. If a sentence does not contain a defined event, we will assign the label NA. Table 3 shows examples of annotated sentences. It is noted that if a sentence contains the same event, our model still treats it as multiple events.

Table 3. Examples of annotated instances.

Sentences	Labels
Armored forces destroyed dozens of Iraqi tanks and personnel carriers in their advance on Baghdad.	Attack, Transport
They got married in 1985.	Marry
We don't know that all foist is back, but at least some of it is.	NA

In this paper, to evaluate the performance of our proposed THEED framework, we compared our model with other advanced methods in the ACE2005 dataset.

- **DMCNN [8]**, extracting word-level and sentence-level features from text automatically using dynamic multipooling layers.
- **JRNN [23]**, a joint framework of bidirectional recurrent neural networks, is used for event extraction.
- **TBNNAM [10]**, which encodes the representation of sentences according to the target event type without triggers.
- **Text2Event [32]** is a sequence-to-structure model that directly learns from text annotation and requires no trigger offsets.
- **DEEB-RNN3 [25]**, which uses chapter-level embedding representation as a reinforcement for sentence-level event detection.
- **OntoED [29]**, a novel framework for event detection with the help of ontology embedding, describing event detection as the process of ontology filling.
- **Query and Extract [9]**, which refines event extraction as a query-and-extract paradigm and leverages rich semantics of event types.

4.2. Experiment Settings

Regarding the settings of the model training process, the Adam [44] optimizer is used, with 100 iterations of training. In our experiment, the dimension of word embedding is

200, entity type and part-of-speech set is 50 and the maximum length of a sequence is 128. In THEED, a dropout rate of 0.2 is used to avoid overfitting, the L2 norm is set to 10-5 and β in the bias term is 1. Furthermore, we dynamically adapt the parameter λ on the dev dataset, and according to the F1 scores in Figure 5, we take the value of λ as 0.3.

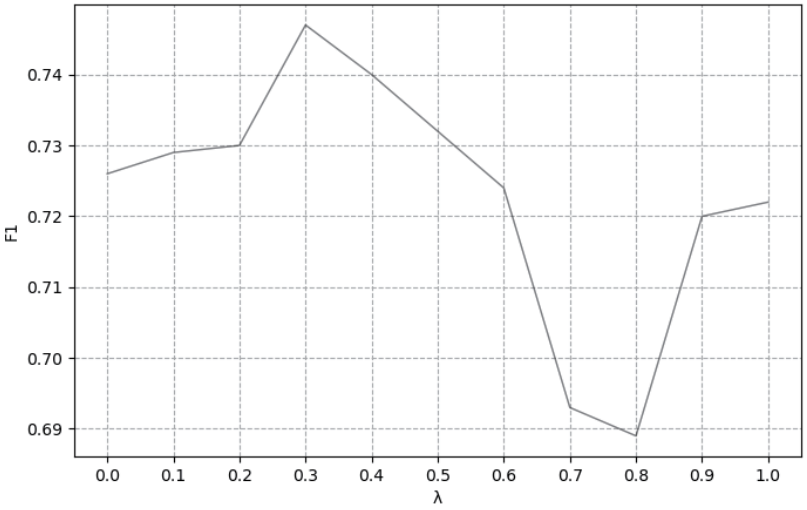


Figure 5. Experimental results on test dataset with different settings of λ.

4.3. Overall Evaluation

In this section, we analyze the results of the proposed framework (see Table 4). They show that the overall performance of this model is better than other baselines. Specifically, we can have the following points from the results.

Table 4. Experimental results of event detection on ACE2005 corpus.

Model	P (%)	R (%)	F1 (%)
DMCNN	75.7	66.0	70.5
JRNN	66.0	73.0	69.3
TBNNAM	76.2	64.5	69.9
TEXT2EVENT	69.6	74.4	71.9
DEEB-RNN3	72.3	75.8	74.0
OntoED	76.6	75.0	76.2
Query-And-Extract	-	-	73.5
THEED (ours)	78.1	71.6	74.7

- Our approach is significantly improved compared with the traditional neural network models DMCNN and JRNN for event extraction methods. It indicates that bi-LSTM can enrich semantic information. Furthermore, we add part-of-speech features that also help the event detection task.
- THEED attains a much better performance improvement (about 4.5% and 2.5% in F1 scores) compared with TBNNAM and TEXT2EVENT, respectively, which do not have access to the trigger annotations or the trigger offsets. The results show that our proposed type hierarchy concept modular component is useful. Compared with the other methods of querying an event type/subtype in the "embedding table" to obtain a single semantic piece of information, the type hierarchy concept can effectively enrich the representation of event types. However, TEXT2EVENT exploits the event schema to guide event generation and uniformly predicts all event types, The nonbalanced distribution of event types will have less impact on the model, so its recall is higher than THEED. However, the model is much more complex than THEED, and the F1 value is lower than THEED

- Although our proposed approach does not use annotated triggers, it achieves improvements over other models, namely DEEB-RNN3 and Query-And-Extract. This is because our method adopts both word-level and context-level representations, which is helpful for the event detection task. On the other hand, DEEB-RNN3 exploits document embedding, which may learn the relationship between event types in the same document, leading to a higher recall than our method. However, the method requires annotated event information.
- THEED can outperform all compared methods except OntoED, which formulates event detection as a process of event ontology population; it enables the utilization and propagation of event knowledge, especially from high-resource to low-resource event types. Despite the lack of propagation and inference of knowledge, our model has a close F1 score to OntoED. However, our method requires fewer manual annotations.
- From the experimental results, we can observe a phenomenon that the recall of the model is relatively lower. This is because if a sentence contains two identical event types, such as "Attack, Attack", we treat them as two different events, while other models treat them as only one event. On the other hand, the nonbalanced distribution of the number of training events also affects the recall of our method.

4.4. Ablation Study

In our proposed method, we undertake an ablation study to evaluate the effect of the word-level and context-level representations and type hierarchy concepts. Firstly, since we perform type hierarchy concept enhancement for each event type, we report our framework's performance without a hierarchy concept and concept confidence. In particular, an average score will replace concept confidence when removing it. Table 5 illustrates that our model is not affected by sparsity, and confidence measures the similarity of the hierarchy concept to the event type. Additionally, the results show that hierarchical concepts have a more significant impact on the model, which proves the effectiveness of our proposed method.

Table 5. Performance of framework without hierarchy concept and hierarchy concept confidence.

Model	P (%)	R (%)	F1 (%)
without hierarchy concept	77.2	68.0	72.3
without confidence	77.3	69.3	73.1

Secondly, we verify the effect of word-level representation, context-level representation, superordinate concepts and subordinate concepts on the model. For example, if we only use word-level representation, the hierarchy concept remains unchanged. As Table 6 shows, all of the modular components display significant benefits to event detection. Specifically, we can see that using a single representation of a sentence will cause a significant drop in results, which demonstrates the effectiveness of two-level representation.

Table 6. Performance of the effect of word-level representation, context-level representation, superordinate concepts and subordinate concepts on the model.

Model	P (%)	R (%)	F1 (%)
only word-level	76.6	68.2	72.2
only context-level	76.7	69.0	72.6
only sup-concept	77.7	69.0	73.1
only sub-concept	77.4	68.6	72.7

To further verify the effect of the hierarchy concept, we replace it with the prototype triggers mentioned in the previous work [9]. Taking the event type Contact:Meet as an example, we finally represent it as Meet *meet reunited retreats*. Table 7 shows experimental results. From the table, we observe that the hierarchy concept achieves better performance,

which is understandable as prototype triggers may be ambiguous triggers for event detection, such as "meet". However, compared with other ablation experiments, the results show some improvement, indicating that semantic enhancement of event type is effective for event detection.

Table 7. Performance of the framework replacing hierarchy concept with prototype triggers.

Model	P (%)	R (%)	F1 (%)
prototype triggers	77.8	70.2	73.8
THEED (ours)	78.1	71.6	74.7

4.5. Case Study

To verify whether the word-level representation modular components work as we designed, we conduct a case study. As Figure 6 shows, we visualize the attention score α^j of our framework on a random sentence: "the army's 3rd infantry division 1st marine division, leading the assault against Baghdad", and its event type is Attack. From the experimental results, we can see that the word "assault" has the highest score, and according to the ranking, we obtain the top K words as follows: *army infantry 1st division the assault against Baghdad*. In this way, we successfully captured the word-level representation associated with the event type while filtering out irrelevant words.

Furthermore, to verify the effect of the type hierarchy concept modular components, we also conduct a case study. As Figure 7 shows, we visualize the final scores of the sentence: "the army's 3rd infantry division 1st marine division, leading the assault against Baghdad" on word-level representation, context-level representation, superordinate concepts and subordinate concepts, where T denotes the correct classification and F denotes incorrect. Due to the small granularity of scores, the difference between heatmaps at the same level is not apparent. However, the scores between correct and incorrect event types are still obvious.

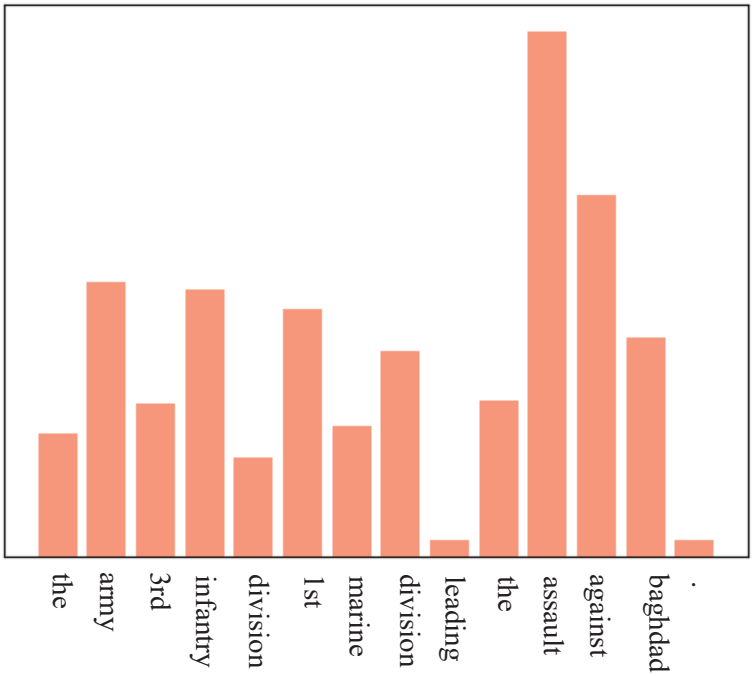


Figure 6. The attention scores for the hidden embeddings of words, which our model learns from a sample instance.

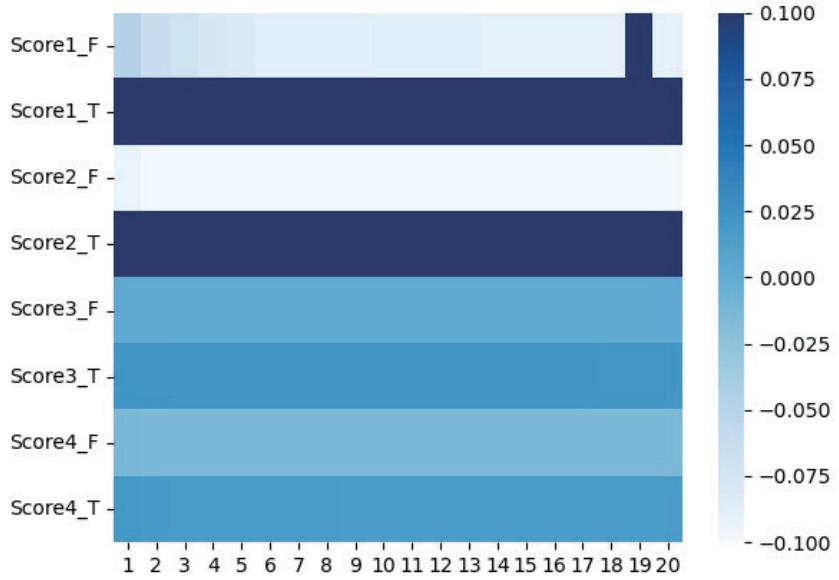


Figure 7. Heatmap for final scores to verify the effect of the type hierarchy concept modular components, and the input sentence is "the army's 3rd infantry division 1st marine division, leading the assault against Baghdad".

4.6. Error Analysis

We analyze the errors according to the event detection results on the test set. On the one hand, our proposed framework does not perform well in low-resource scenarios. For event types that frequently appear in the training corpus, such as *Attack Die*, the model can achieve better performance. As for the lower frequency event types, such as *nominate Merge-Org* event types, the results of F1 are not good. On the other hand, we can see that the recall of our experimental results is low from Table 4, which indicates that our framework still learns limited features. Although we use an external knowledge graph to extend the event types, the effect is still restricted. We will try to use more effective methods to solve this problem in our future work.

5. Conclusions and Future Work

This paper proposes a novel event detection framework with enahnced type hierarchy called THEED. We construct a type hierarchy concept module using Probase to enhance the semantic representation of event types, and we assign hierarchy concepts with confidence. In addition, dividing the input into word-level and context-level representations helps the model extract different features. Experimental results demonstrate that our approach performs better on event detection without triggers. Notably, our proposed method achieves competitive performances compared with state-of-the-art trigger-based models. In the future, we will explore better representations with the help of external knowledge graphs to improve the accuracy and generalizability of event detection without triggers. We also plan to complete the event extraction task without triggers.

Author Contributions: Conceptualization, Y.Y., Z.L., J.G. and F.G.; methodology, Y.Y. and J.G.; software, Y.Y.; validation, Y.Y., J.G. and F.G.; formal analysis, Y.Y.; investigation, J.G. and F.G.; writing—original draft preparation, Y.Y.; writing—review and editing, J.G. and F.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the National Natural Science Foundation of China grant number U1836118, Key Laboratory of Rich Media Digital Publishing, Content Organization and Knowledge Service grant number ZD2022-10/05, and National key research and development program grant number 2020AAA0108500.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this approach are available from the public datasets. These data are available at: https://catalog.ldc.upenn.edu/LDC2006T06 (accessed on 15 February 2006).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, Z.; Wang, X.; Han, X.; Lin, Y.; Hou, L.; Liu, Z.; Li, P.; Li, J.; Zhou, J. CLEVE: Contrastive Pre-training for Event Extraction. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP), Virtual Event, 1–6 August 2021; Volume 1, pp. 6283–6297.
2. Liu, J.; Chen, Y.; Liu, K.; Bi, W.; Liu, X. Event Extraction as Machine Reading Comprehension. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020), Online, 16–20 November 2020; pp. 1641–1651.
3. Hettiarachchi, H.; Adedoyin-Olowe, M.; Bhogal, J.; Gaber, M.M. Embed2Detect: Temporally clustered embedded words for event detection in social media. *Mach. Learn.* **2022**, *111*, 49–87. [CrossRef]
4. Liang, X.; Cheng, D.; Yang, F.; Luo, Y.; Qian, W.; Zhou, A. F-HMTC: Detecting Financial Events for Investment Decisions Based on Neural Hierarchical Multi-Label Text Classification. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI), Yokohama, Japan, 11–17 July 2020; pp. 4490–4496.
5. Nikiforos, M.N.; Vergis, S.; Stylidou, A.; Augoustis, N.; Kermanidis, K.L.; Maragoudakis, M. Fake News Detection Regarding the Hong Kong Events from Tweets. In Proceedings of the AIAI 2020: Artificial Intelligence Applications and Innovations. AIAI 2020 IFIP WG 12.5 International Workshops, Neos Marmaras, Greece, 5–7 June 2020; Volume 585, pp. 177–186.
6. Hu, H.; Wang, C.; Dai, J.; Liu, M. Social Emergency Event Judgement based on BiLSTM-CRF. In Proceedings of the 19th Chinese National Conference on Computational Linguistics Haikou, China, 30 October–1 November 2020; Chinese Information Processing Society of China: Haikou, China, 2020; pp. 667–675.
7. Wang, X.; Han, X.; Liu, Z.; Sun, M.; Li, P. Adversarial Training for Weakly Supervised Event Detection. In 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, 2–7 June 2019; Volume 1, pp. 998–1008.
8. Chen, Y.; Xu, L.; Liu, K.; Zeng, D.; Zhao, J. 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), Beijing, China, 26–31 July 2015; Volume 1, pp. 167–176.
9. Wang, S.; Yu, M.; Chang, S.; Sun, L.; Huang, L. Query and Extract: Refining Event Extraction as Type-oriented Binary Decoding. In Proceedings of the Findings of the Association for Computational Linguistics (ACL 2022), Dublin, Ireland, 22–27 May 2022; pp. 169–182.
10. Liu, S.; Li, Y.; Zhang, F.; Yang, T.; Zhou, X. Event Detection without Triggers. In 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, 2–7 June 2019; Volume 1, pp. 735–744.
11. Zeng, Y.; Feng, Y.; Ma, R.; Wang, Z.; Yan, R.; Shi, C.; Zhao, D. Scale Up Event Extraction Learning via Automatic Training Data Generation. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, LA, USA, 2–7 February 2018; pp. 6045–6052.
12. Zheng, S.; Cao, W.; Xu, W.; Bian, J. Doc2EDAG: An End-to-End Document-level Framework for Chinese Financial Event Extraction. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019), Hong Kong, China, 3–7 November 2019; pp. 337–346.
13. Huang, H.; Wang, Y.; Feng, C.; Liu, Z.; Zhou, Q. Leveraging Conceptualization for Short-Text Embedding. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 1282–1295. [CrossRef]
14. Wang, Y. Leveraging Lexical Common-Sense Knowledge for Boosting Bayesian Modeling. In Proceedings of the Natural Language Processing and Chinese Computing—10th CCF International Conference (NLPCC 2021), Qingdao, China, 13–17 October 2021; Springer: Berlin/Heidelberg, Germany, 2021; Volume 13028, pp. 643–651.
15. Li, J.; Huang, X.; Gao, Y.; Liu, J.; Zhang, R.; Zhao, J. Distant Supervised Relation Extraction Based on Sentence-Level Attention with Relation Alignment. In Proceedings of the 8th International Conference on Artificial Intelligence and Security (ICAIS 2022), Qinghai, China, 15–20 July 2022; Springer: Berlin/Heidelberg, Germany, 2022; Volume 13338, pp. 142–152.
16. Wang, C.; Zhang, Y.; Guo, J.; Gao, S.; Yu, Z. Event detection without trigger words incorporating syntactic information. *J. Comput. Appl.* **2021**, *41*, 3534–3539.
17. He, X.; Tai, P.; Lu, H.; Huang, X.; Ren, Y. A biomedical event extraction method based on fine-grained and attention mechanism. *BMC Bioinform.* **2022**, *23*, 308. [CrossRef] [PubMed]
18. Zhang, S.; Li, Y.; Li, S.; Yan, F. Bi-LSTM-CRF Network for Clinical Event Extraction With Medical Knowledge Features. *IEEE Access* **2022**, *10*, 110100–110109. [CrossRef]

19. Ahn, D. The stages of event extraction. In Proceedings of the Workshop on Annotating and Reasoning about Time and Events, Sydney, Australia, 23 July 2006; pp. 1–8.
20. Liao, S.; Grishman, R. Using Document Level Cross-Event Inference to Improve Event Extraction. In Proceedings of the ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 11–16 July 2010; pp. 789–797.
21. Qin, Y.; Zhang, Y.; Zhang, M.; Zheng, D. Feature-Rich Segment-Based News Event Detection on Twitter. In Proceedings of the Sixth International Joint Conference on Natural Language Processing (IJCNLP 2013), Nagoya, Japan, 14–18 October 2013; pp. 302–310.
22. Li, Q.; Ji, H.; Huang, L. Joint Event Extraction via Structured Prediction with Global Features. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013), Sofia, Bulgaria, 4–9 August 2013; Volume 1, pp. 73–82.
23. Nguyen, T.H.; Cho, K.; Grishman, R. Joint Event Extraction via Recurrent Neural Networks. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2016), San Diego, CA, USA, 12–17 June 2016; pp. 300–309.
24. Feng, X.; Qin, B.; Liu, T. A language-independent neural network for event detection. *Sci. China Inf. Sci.* **2018**, *61*, 092106:1–092106:12. [CrossRef]
25. Zhao, Y.; Jin, X.; Wang, Y.; Cheng, X. Document Embedding Enhanced Event Detection with Hierarchical and Supervised Attention. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018), Melbourne, Australia, 15–20 July 2018; Volume 2, pp. 414–419.
26. Hu, D.; Wang, M.; Gao, F.; Xu, F.; Gu, J. Knowledge Representation and Reasoning for Complex Time Expression in Clinical Text. *Data Intell.* **2022**, *4*, 573–598. [CrossRef]
27. Li, D.; Huang, L.; Ji, H.; Han, J. Biomedical Event Extraction based on Knowledge-driven Tree-LSTM. In 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, 2–7 June 2019; Volume 1, pp. 1421–1430.
28. Wang, M.; Gao, F.; Gu, J. Process-Aware Dialogue System With Clinical Guideline Knowledge. *Int. J. Web Serv. Res.* **2022**, *19*, 1–22. [CrossRef]
29. Deng, S.; Zhang, N.; Li, L.; Chen, H.; Tou, H.; Chen, M.; Huang, F.; Chen, H. OntoED: Low-resource Event Detection with Ontology Embedding. In 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), Virtual Event, 1–6 August 2021; Volume 1, pp. 2828–2839.
30. Xu, W.; Zhang, W.; Wang, D. Event detection without trigger words on movie scripts. *Proc. SPIE* **2020**, *11584*, 342–347.
31. Hsu, I.; Huang, K.; Boschee, E.; Miller, S.; Natarajan, P.; Chang, K.; Peng, N. DEGREE: A Data-Efficient Generation-Based Event Extraction Model. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2022), Seattle, WA, USA, 10–15 July 2022; pp. 1890–1908.
32. Lu, Y.; Lin, H.; Xu, J.; Han, X.; Tang, J.; Li, A.; Sun, L.; Liao, M.; Chen, S. Text2Event: Controllable Sequence-to-Structure Generation for End-to-end Event Extraction. In 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), Virtual Event, 1–6 August 2021; Volume 1, pp. 2795–2806.
33. Zeng, D.; Liu, K.; Lai, S.; Zhou, G.; Zhao, J. Relation Classification via Convolutional Deep Neural Network. In Proceedings of the 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers (COLING 2014), Dublin, Ireland, 23–29 August 2014; pp. 2335–2344.
34. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. In Proceedings of the 1st International Conference on Learning Representations (ICLR 2013), Scottsdale, AZ, USA, 2–4 May 2013.
35. Gururangan, S.; Marasović, A.; Swayamdipta, S.; Lo, K.; Beltagy, I.; Downey, D.; Smith, N.A. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. *arXiv* **2020**, arXiv:2004.10964.
36. Graves, A. Long short-term memory. In *Supervised Sequence Labelling with Recurrent Neural Networks*; Springer: Berlin/Heidelberg, Germnay, 2012; pp. 37–45.
37. Schuster, M.; Paliwal, K.K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681. [CrossRef]
38. Liu, J.; Chen, Y.; Xu, J. Saliency as Evidence: Event Detection with Trigger Saliency Attribution. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022), Dublin, Ireland, 22–27 May 2022; Volume 1, pp. 4573–4585.
39. Liao, J.; Sun, F.; Gu, J. Combining Concept Graph with Improved Neural Networks for Chinese Short Text Classification. In Proceedings of the Semantic Technology—9th Joint International Conference (JIST 2019), Hangzhou, China, 25–27 November 2019; Springer: Berlin/Heidelberg, Germnay, 2019; Volume 1157, pp. 205–212.
40. Wu, W.; Li, H.; Wang, H.; Zhu, K.Q. Probase: A probabilistic taxonomy for text understanding. In Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2012), Scottsdale, AZ, USA, 20–24 May 2012; pp. 481–492.
41. Yang, H.; Chen, Y.; Liu, K.; Xiao, Y.; Zhao, J. DCFEE: A Document-level Chinese Financial Event Extraction System based on Automatically Labeled Training Data. In Proceedings of the Proceedings of ACL 2018, Melbourne, Australia, 15–20 July 2018; pp. 50–55.

42. Chen, Y.; Yang, H.; Liu, K.; Zhao, J.; Jia, Y. Collective Event Detection via a Hierarchical and Bias Tagging Networks with Gated Multi-level Attention Mechanisms. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 1267–1276.
43. Walker, C.; Strassel, S.; Medero, J.; Maeda, K. ACE 2005 multilingual training corpus. *Linguist. Data Consortium Phila.* **2006**, *57*, 45.
44. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015), San Diego, CA, USA, 7–9 May 2015.