• 计算机工程与应用 •

Exploiting Document-Level Information to Enhance Event Detection Combined with Semantic Space

LUO Jinshang[1], SHI Xin[2], WU Jie[2], and HOU Mengshu[2*]

(1. Information Center, University of Electronic Science and Technology of China Chengdu 611731;

2. School of Computer Science and Engineering, University of Electronic Science and Technology of China Chengdu 611731)

Abstract Event detection (ED) is a fundamental task of event extraction, which aims to detect triggers in text and determine their event types. Most existing methods regard event detection as a sentence-level classification problem, ignoring the correlations between events in different sentences. A novel event detection framework, named document embedding networks combined with semantic space (DENSS), is proposed in this paper. The document-level information is utilized to alleviate semantic ambiguity and enhance contextual understanding. Specifically, the representations of event types and triggers are obtained through an off-the-shelf pre-trained model and a designed multi-level attention mechanism. Then the feature vectors of event types and triggers are mapped into a shared semantic space, where the distance represents the correlation of different events. The experimental results on the benchmark dataset demonstrate that our method outperforms most existing methods, and justify the effectiveness of document-level information with shared semantic space.

Key words BERT; document-level information; event detection; semantic space

利用文档级信息结合语义空间加强事件检测

罗锦尚[1]，时 昕[2]，吴 洁[2]，侯孟书[2*]

(1. 电子科技大学信息中心 成都 611731；2. 电子科技大学计算机科学与工程学院 成都 611731)

【摘要】 事件检测 (ED) 是事件抽取的一项基础任务，旨在检测事件触发器并进行分类。现有事件检测方法主要基于句子级信息，忽略了句子间的事件相关性。文档级信息有助于减轻语义歧义与加强上下文理解，为此，提出一种新颖的事件检测框架，命名为 document embedding networks combined with semantic space (DENSS)。首先，利用了预训练语言模型，分别表示具有丰富语义信息的事件类型与事件触发器；设计一种多层次注意力机制，用以捕获句子级和文档级信息；映射事件类型与事件触发器的特征向量到一个共享的语义空间，事件的相关性被表示为事件嵌入的距离；最后，基于基准数据集进行了验证，结果表明该方法优于大部分已有的方法，以及具有共享语义空间的文档级信息对于加强事件检测的有效性。

关 键 词 BERT； 文档级信息； 事件检测； 语义空间

中图分类号 TP183 **文献标志码** A doi:10.12178/1001-0548.2021304

Event detection (ED) is a crucial task of event extraction (EE), which aims to identify event triggers from text and classify them into corresponding event types. The event trigger is the word or phrase that can clearly indicate the existence of an event in a sentence. According to the automatic context extraction (ACE) 2005 dataset, which is widely applied to the ED task, there are 8 event types and 33 subtypes, such as “Attack”, “Transport”, “Meet” etc. Take the following sentences as examples:

S1: He has **died** of his wounds after being **shot**.

S2: An American tank **fired** on the Palestine hotel.

S3: Another veteran war correspondent is being **fired** for his controversial conduct in Iraq.

Received date: 2021 − 10 − 19; Revised date: 2022 − 01 − 09

收稿日期：2021 − 10 − 19；修回日期：2022 − 01 − 09

Biography: LUO Jinshang was born in 1981, male, engineer, his research interests include machine learning and natural language processing.

作者简介：罗锦尚 (1981 −)，男，工程师，主要从事机器学习、自然语言处理等方面的研究.

*Corresponding author: HOU Mengshu, E-mail: mshou@uestc.edu.cn.

*通信作者：侯孟书，E-mail：mshou@uestc.edu.cn

An ideal ED model is expected to recognize two events: a "Die" event triggered by the trigger word "died" and an "Attack" event triggered by "shot" in S1.

The difficulty of the ED task lies in the diversity and ambiguity of natural language expression. On the one hand, there are a variety of expressions that belong to the same event type. In S1, "shot" triggers an "Attack" event, and "fired" also triggers the same event type in S2. On the other hand, the same trigger can denote different events. In S3, "fired" can trigger an "Attack" event or an "End-Position" event. Because of the ambiguity, a traditional approach may mislabel "fired" with "Attack" according to the word "war" with sentence-level information. However, in the same document, other sentences like "NBC is terminating freelancer reporter Peter Arnett for statements he made to the Iraqi media." could provide the clue that "fired" triggers an "End-Position" event. Up to 57% of the event triggers are ambiguous in the ACE 2005 dataset[1]. Thus, how to solve the ambiguity of event trigger has become an important problem in ED task.

ED is a booming and challenging task in NLP. The dominant approaches for ED adopt deep neural networks to learn effective features for the input sentences. Most existing methods either generally focus on sentence-level context, or ignore the correlations between events, such as semantic correlation information. Many methods[2-3] mainly exploit sentence-level features that lack a summary of the document. Sometimes sentence-level information is insufficient to address the ambiguity of event trigger, such as the event trigger "fired" in S3. Some document-level models have been proposed to leverage global context [4-6]. However, these methods extract features of the entire document, which are coarse-grained features for event classification. Actually, by means of processing context more effectively, the model's performance can be improved.

The semantic correlations between different events exist objectively and pervasively, and they are manifested in several aspects. Initially, different event types have some semantic relevance. For instance, compared with the "Transport" event, the "Attack" event and the "Injure" event are semantically closer. Belonging to the same parent event type, different subtypes have certain semantic correlations. "Be-Born" and "Marry" belong to the same parent event type "Life", which can reveal more collective features. They are more likely to co-occur in the same document. Furthermore, different event triggers have some semantic correlations in the same document, such as event trigger "shot" and "died" in S1. The events mentioned in the same document tend to be semantically coherent. As pointed out by Ref. [5], many events usually co-occur in the same document. According to the ACE 2005 dataset, the top 5 event types that accompany with "Attack" event in the same sentence are as follows: Attack, Die, Transport, Injure and Meet. Eventually, there is similar semantics between the event trigger and its corresponding event type. The event type word indicates the fundamental semantic information and reveals common features, and the event trigger word has extended semantic information with a more specific context. Suppose we replace the trigger word with its corresponding event type word, the semantics of the whole sentence will not change much. Thus, how to model the semantic correlation information between event types and event triggers becomes a challenge to be overcome.

Existing methods generally use the one-hot label, which classifies the event type with the 0/1 label. Despite the simplicity, it regards multiple events in the same document as independent ones, and therefore it is difficult to accurately represent the correlations between different event types.

In this paper, we propose document embedding networks with shared semantics space (DENSS) to address the aforementioned problems. To learn the event correlations, we use bidirectional encoder representations from transformers (BERT) to obtain event type representations and map them into a semantic space, where the more relevant event types are, the closer they stay. We apply BERT again to acquire the representation of each word with document-level and sentence-level information via gated attention, project the representation of each event trigger into the same semantic space, and choose the label of the closest event type.

In summary, the contributions of this paper are as follows: 1) We study the event correlations problem and propose a novel ED framework, which utilizes BERT for capturing document-level and sentence-level information. 2) We employ a shared semantic space to represent event types and event triggers, which minimizes the distance between each event trigger and its corresponding type. Experiment results on the ACE 2005 dataset verify the effectiveness of our approach.

1 Approach

1.1 Task Description

The goal of ED consists of identifying event triggers (trigger identification) and classifying them into corresponding event types (trigger classification). According to the ACE 2005 dataset, an event is defined as a specific occurrence involving one or more participants. The event trigger is the main word or phrase that can most clearly express the occurrence of an event. As shown in Table 1, the ACE 2005 dataset defines 8 event types and 33 subtypes. Each event subtype has its specific semantic information and different event subtypes have certain semantic correlations.

Table 1 Some Event Types and Subtypes of the ACE 2005 Dataset

Event Type	Event Subtype
Life	Be-Born、Marry、Divorce、Injure、Die
Movement	Transport
Personnel	Start-Position、End-Position、Elect、Nominate
Conflict	Demonstrate、Attack
Business	Merge-Org、Start-Org、End-Org、Declare-Bankruptcy

Formally, given a training set $D=\{d_1,d_2,\cdots d_i,\cdots d_l\}$, where l is the number of documents in training set, and a document $d=\left\{s_1,s_2,\cdots s_j,\cdots s_m\right\}$, where m is the number of sentences in the document d, the j-th sentence can be represented as $s_j=\{w_{j1}, w_{j2},\cdots w_{jk},\cdots w_{jn}\}$, where n is the number of words in sentence s_j.

1.2 Overview

We formalize ED as a multi-label sequence tagging problem. We assign a tag for each word to indicate whether it triggers the specific event. We adopt the “BIO” tags schema. Tags “B” and “I” represent the position of the word in a trigger to solve the problem that a trigger contains multiple words such as “take away”, “go to” and so on.

Figure 1 describes the architecture of DENSS, which primarily involves the following four components: 1) Event embedding, which learns correlations between event types through BERT; 2) Word embedding, which exploits BERT and gated attention to gain semantic information of words; 3) Trigger identification, which identifies the event triggers; 4) Trigger classification, which classifies the event triggers to corresponding types.

1.3 Event Embedding

To enrich the contextual information of event type words, we replace each trigger word in the sentence with the corresponding event type word. For instance, sentence S1 is transformed into another sentence “He has **die** of his wounds after being **attack**”. Sentence S3 is converted into a new sentence “Another veteran war correspondent is being **end-position** for his controversial conduct in Iraq”. Contextualized embedding produced by pre-trained language models[7] has been proved to be capable of modeling context beyond the sentence boundary and improving performance on a variety of tasks. Pre-trained bi-directional transformer models such as BERT can better capture long-distance dependencies as compared with Recurrent Neural Network (RNN) architecture. These newly replaced sentences are fed into BERT, and the last layer's hidden vectors of BERT are set as the words' embedding. Let E_i be the event embedding corresponding to the i-th event type word. For simplicity, we calculate the average of all representations to get the final representation for the event type word, which appears many times in the training sentences. The ACE 2005 dataset defines 33 subtypes. According to “BIO” tags schema, we finally obtain 67 representations of the event type words, as $E=\{E_1,E_2,\cdots E_y,\cdots E_{67}\}$, and map the feature vectors E_y into a shared semantic space.

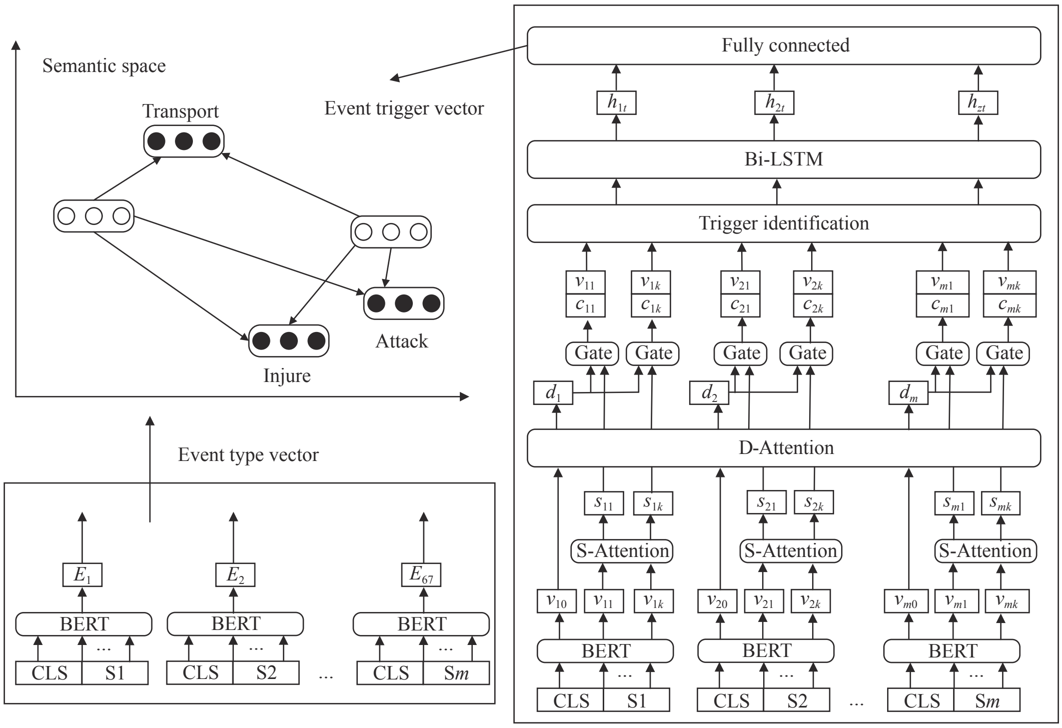

Fig. 1　The Architecture of the DENSS Model

To give an intuitive illustration, the different event correlations are shown in Figure 2.

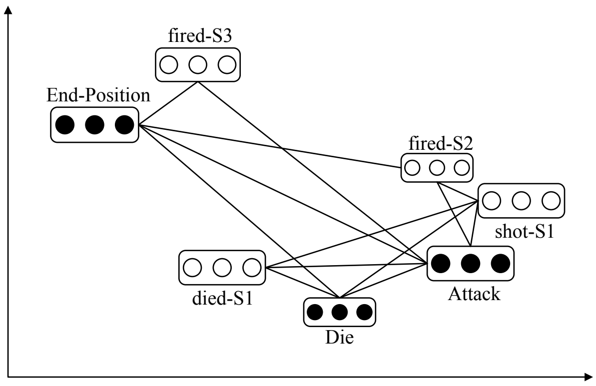

Fig. 2　Event Correlation

In this figure, the solid circle denotes the event type and event type vector. The empty circle denotes the event trigger and event trigger vector.

1.4 Word Embedding

1.4.1 Word-level Embedding

Given a document $d=\{s_1,s_2,\cdots s_j,\cdots s_m\}$, the j-th sentence can be represented as token sequence $s_j=\{w_{j1},w_{j2},\cdots w_{jk},\cdots w_{jn}\}$. Special tokens [CLS] and [SEP] are placed at the start and end of the sentence, as $\{[\text{CLS}],w_{j1},w_{j2},\cdots w_{jk},\cdots w_{jn},[\text{SEP}]\}$. BERT can create token embedding, segment embedding, position embedding automatically, and concatenate these embeddings as the input of the next layer. For each word w_{jk}, we select the feature vector from the last layer of BERT as word embedding v_{jk}. The sentence s_j is represented as $\{v_{j1},v_{j2},\cdots v_{jk},\cdots v_{jn}\}$. By considering the embedding of token [CLS] as the sentence embedding, simultaneously we obtain the sentence embedding v_{j0}, which corresponds to token [CLS].

1.4.2 Sentence-level Embedding

The sentence-level attention mechanism is utilized to extract the important clues at sentence level. For each word w_{jk}, we employ the attention mechanism to calculate its relatedness with all other words in the sentence. r_s^t is the relatedness between the k-th word representation v_{jk} and the t-th word representation v_{jt}:

$$\boldsymbol{r}_s^t=\tanh(v_{jk}\boldsymbol{W}_{sa}v_{jt}^{\mathrm{T}}+b_{sa}) \tag{1}$$

where $\boldsymbol{W}_{sa}$ is the weight matrix and b_{sa} is the bias term. $\boldsymbol{r}_s^t$ is then normalized to obtain scalar attention weight α_s^t:

$$\alpha_s^t=\frac{\exp(r_s^t)}{\sum_{x=1}^{n}\exp(r_s^x)} \tag{2}$$

For each word w_{jk}, its sentence-level semantic information $\boldsymbol{s}_{jk}$ is calculated by:

$$\boldsymbol{s}_{jk} = \sum_{t=1}^{n} \alpha_s^t v_{jt} \tag{3}$$

1.4.3 Document-level Embedding

Similar to sentence-level attention, we utilize the document-level attention mechanism for capturing the significant clues at the document level. For each sentence s_j, we employ the attention mechanism to calculate its relatedness with all other sentences in the document. $\boldsymbol{r}_d^t$ is the relatedness between the j-th sentence representation v_j and the t-th sentence representation v_t:

$$\boldsymbol{r}_d^t = \tanh(v_j \boldsymbol{W}_{da} v_t^{\mathrm{T}} + b_{da}) \tag{4}$$

where $\boldsymbol{W}_{da}$ is the weight matrix and b_{da} is the bias term. $\boldsymbol{r}_d^t$ is then normalized to obtain scalar attention weight α_d^t:

$$\alpha_d^t = \frac{\exp(r_d^t)}{\sum_{x=1}^{m} \exp(r_d^x)} \tag{5}$$

For each sentence s_j, its document-level semantic information d_j is calculated by:

$$d_j = \sum_{t=1}^{m} \alpha_d^t v_t \tag{6}$$

1.4.4 Gated Fusion

Inspired by the gated multi-level attention mechanisms[5], we apply a fusion gate to dynamically incorporate sentence-level information s_{jk} and document-level information d_j for the k-th word w_{jk} in the j-th sentence s_j of the document d. The fusion gate g_k is designed to control how information should be integrated, which is calculated by:

$$g_k = \sigma(\boldsymbol{W}_g[s_{jk}, d_j] + b_g) \tag{7}$$

where $\boldsymbol{W}_g$ is the weight matrix, b_g is the bias term, and σ is the sigmoid function. Hence, the contextual representation of the word w_{jk} with both sentence-level information and document-level information is calculated by:

$$c_{jk} = (g_k \otimes s_{jk}) + ((1 - g_k) \otimes d_j) \tag{8}$$

where $\otimes$ denotes element-wise multiplication. We concatenate the contextual representation c_{jk} and word embedding v_{jk} to acquire the final word representation $e_{jk} = [c_{jk}, v_{jk}]$.

1.5 Trigger Identification

We model trigger identification task as a binary classification problem and annotate the trigger with label 1 while the others with label 0. The final word representation e_{jk} is fed into a binary classifier to decide whether it is a trigger.

1.6 Trigger Classification

The bidirectional long short term memory (Bi-LSTM) has been proven effective to capture the semantic information of words[8]. To learn the correlations of different triggers, we filter out the words that are not triggers and assign all triggers of the document into a sequence. Let $\{e_{1t}, e_{2t}, \cdots e_{jt}, \cdots e_{zt}\}$ refer to the trigger representation sequence in document d, where e_{jt} is the real-value vector. Then, we feed the sequence into Bi-LSTM to fuse the contextual information of the triggers with document-level information. The forward LSTM generates the forward hidden vector sequence $\{\overrightarrow{h_{1t}}, \overrightarrow{h_{2t}}, \cdots \overrightarrow{h_{jt}}, \cdots \overrightarrow{h_{zt}}\}$ and the backward LSTM generates the backward hidden vector sequence $\{\overleftarrow{h_{1t}}, \overleftarrow{h_{2t}}, \cdots \overleftarrow{h_{jt}}, \cdots \overleftarrow{h_{zt}}\}$. Thus, we acquire the trigger feature sequence $\{e_1, e_2, \cdots e_x, \cdots e_z\}$ where $e_x = [\overrightarrow{h_x}, \overleftarrow{h_x}]$ by concatenating the forward and backward hidden states from the Bi-LSTM. A fully connected layer behind Bi-LSTM is added to map the feature vector e_x of the trigger into the antecedent semantic space. Inspired by Refs. [9-10], we exploit cosine similarity to measure the distance between the current trigger and all event types, and choose the label of the closest event type as the label of the trigger.

We adopt cross-entropy loss as the loss function in trigger identification and hinge loss in trigger classification. The hinge loss which is widely used for maximum-margin classification, aims to separate the correct and incorrect predictions with a margin larger than a pre-defined constant. For each trigger x, we name the corresponding event type y as positive and the other types as negative. We construct the hinge ranking loss:

$$L(x, y) = \sum_{i \in Y, i \neq y} \max\{0, b - C_{x,y} + C_{x,i}\} \tag{9}$$

$$C_{x,y} = \cos(e_x, E_y) \tag{10}$$

where y is the corresponding event type of x, Y is the event type set, i is the other event type for x from Y, and b is the margin. The function cos calculates the cosine similarity between the feature vector e_x of the trigger x and the feature vector E_y of the event type y.

2 Experiments

2.1 Dataset and Evaluation Metrics

We conduct experiments on the ACE 2005 dataset. For comparison, we create the same test set with 40 documents, the development set of 30 documents, and the training set of the remaining 529, the same as previous works[2-3]. We adopt the formal ACE evaluation criteria with Precision (P), Recall (R) and F measure (F_1) to evaluate the model.

2.2 Hyper-parameter Setting

Hyper-parameters are tuned on the development set. We employ BERT-base model, which generates 768 dimensional word embedding. We set the dimension of hidden vector as 768, the dimension of semantic space as 768, and margin b as 0.1. We adopt the Adam optimizer for training with a learning rate of 2×10^{-5}.

2.3 Baselines

In order to evaluate our model, we compare it with a comprehensive set of baselines and representative models, including:

1) DMCNN builds the dynamic multi-pooling convolutional neural network to learn sentence-level features[2].

2) JRNN exploits the bidirectional RNN to capture sentence-level features for event extraction[3].

3) GCN-ED applies Graph Convolutional Network (GCN) to model dependency tree for extracting event information[11].

4) DEEB-RNN utilizes document embedding and hierarchical supervised attention mechanism[4].

5) HBTNGMA uses hierarchical and bias tagging networks to detect multiple events[5].

6) PLMEE employs BERT to create labeled data for promoting event extraction[12].

7) DMBERT+Boot utilizes BERT to generate more training data for ED[13].

8) EE-GCN exploits syntactic structure and typed dependency label information to perform ED[14].

2.4 Results

Experimental results are shown in Table 2. From the table, we can observe that our proposed DENSS model achieves the best F_1 score for trigger classification among all the compared methods.

Table 2 Trigger Classification Performance (%) on the ACE 2005 Dataset

Method	P	R	F_1
DMCNN	75.6	63.6	69.1
JRNN	66.0	73.0	69.3
GCN-ED	77.9	68.8	73.1
DEEB-RNN	72.3	75.8	74.0
HBTNGMA	77.9	69.1	73.3
PLMEE	**81.0**	80.4	80.7
DMBERT+Boot	77.9	72.5	75.1
EE-GCN	76.7	78.6	77.6
DENSS	80.6	**82.2**	**81.3**

Compared with DMCNN and JRNN, our method significantly outperforms them. The reason is that DMCNN and JRNN only extract sentence-level information, while our method exploits multi-level information. It indicates that the document-level information is indeed beneficial to ED task. In contrast to DEEB-RNN and HBTNGMA, our method gains great improvement. It's because that DEEB-RNN and HBTNGMA learn document-level information but do not capture rich semantic information. However our method applies pre-trained language model BERT to acquire semantic information of words and employs the semantic space to represent the semantic correlations of different event types. As compared with PLMEE and DMBERT+Boot, our method achieves more desirable performance. PLMEE and DMBERT+Boot use BERT to create training data and promote event extraction, whereas our method fuses multi-level information to represent features of words with rich semantic information. Compared with GCN-ED and EE-GCN, our method is also superior. GCN-ED and EE-GCN adopt GCN with the syntactic information to capture the event information, but the syntactic information is still limited at sentence level. Our method learns the embedding of the document through the hierarchical attention mechanisms, which indicates that multi-level semantic information is conducive to ED task.

2.5 Ablation Study

In this section, we focus on the effectiveness of crucial components in our DENSS model with the ablation study. We examine the following models: 1) EE: to study whether the event embedding contributes to improving the performance, we substitute the one-hot label for the event embedding. As a result, the F_1 score drops by 6.4% absolutely, which demonstrates that the event embedding is beneficial to represent the semantic correlations. 2) SATT: to prove the contribution of the sentence-level attention, we remove it. As can be seen from Table 3, the F_1 score drops by 2.7%, which verifies that the sentence-level information provides important clues. 3) DATT: removing the document-level attention update model hurts the performance by 2.1%, which proves that the document-level information is helpful to enhance the performance. 4) GATE: when we calculate the average of sentence-level information and document-level information instead of the fusion gate, the F_1 score decreases by 1.5%, which indicates that the fusion gate dynamically incorporates multi-level semantic information. 5) Bi-LSTM: Bi-LSTM is removed from the model and the result score decline by 1.8%, which again verifies the effectiveness of the document-level information.

Table 3 The Ablation Study of DENSS

Method	F_1
DENSS	81.3
EE	74.9
SATT	78.6
DATT	79.2
GATE	79.8
Bi-LSTM	79.5

Notes: EE is short for event embedding. SATT is for sentence-level attention. DATT is for document-level attention.

From these ablations, we have the following observations: 1) All crucial components are beneficial to the DENSS model, as removing any component degrades the performance significantly. 2) Compared with others, DENSS-EE substituting the one-hot label for the event embedding hurts the performance deeply. We inference that semantic correlations among event types can propagate more knowledge. 3) As compared with DENSS-DATT, DENSS-SATT has greater performance degradation. It illustrates that the sentence-level information provides more signals than the document-level information commonly. 4) The sentence-level information and document-level information are complementary to the feature representation, and the semantic correlation information is conducive to enhancing ED.

2.6 Case Study

In the section, we present the visualization for the role of the attention mechanism, to validate whether the attention works as we designed. Figure 3 shows the example of the scalar attention weight α learned by our model. In this case, “delivered” triggers a “Phone-Write” event. Our model captures the clue “couriers delivered the letters” and assigns it with a large attention weight. The contextual information plays important role in disambiguating “delivered”, and the words “couriers” and “letters” provide the evidence to predict that “delivered” triggers a “Phone-Write” event.

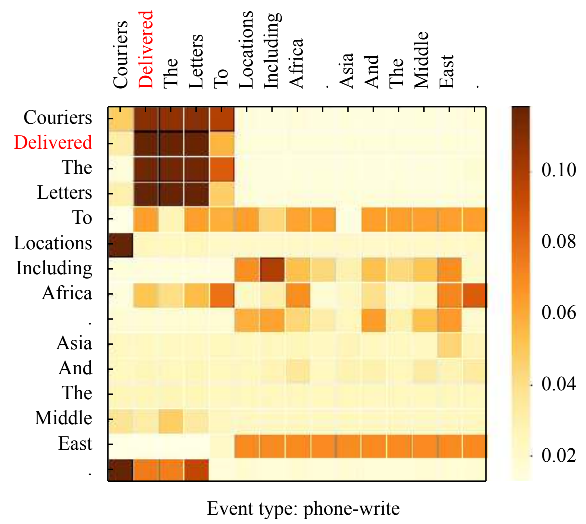

Fig. 3 Visualization for the Role of the Sentence-Level Attention Mechanism. The heat map expresses the contextual attention weight, which represents the relatedness of the corresponding word pair.

Figure 4 shows that the document-level information contributes to improving the performance. We observe that some sentences with the triggers in

Table 4 obtain greater attention weight than others. The triggers "convicted", "killed" and "murdering" in the same document tend to be semantically coherent. It indicates that document-level attention can capture the significant clues at the document level to alleviate semantic ambiguity.

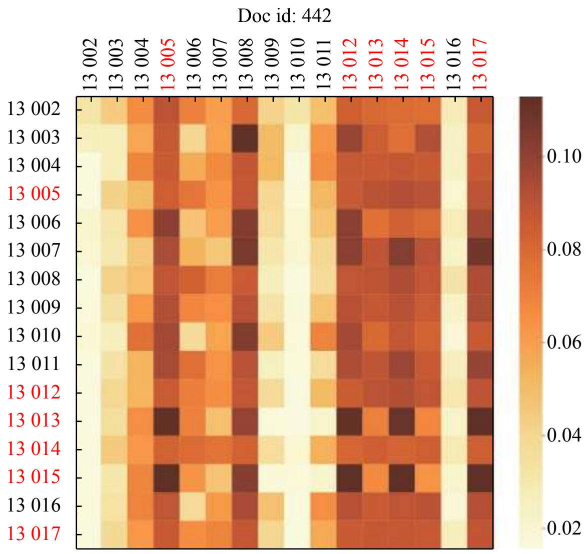

Fig. 4　Visualization for the Role of the Document-Level Attention Mechanism. The heat map expresses the contextual attention weight, which represents the relatedness of the corresponding sentence pair.

Table 4　Example of the Document

Number	Sentence
sent_13002	Kaine on Death and Taxes
⋮	…
sent_13004	Choir boy Tim Kaine is a political moderate informed by his Catholic beliefs
⋮	…
sent_13012	Who other than a left-wing liberal would agree to represent [Sentence] a two-time murderer for free, to try and keep him from getting the death [Die] penalty.
⋮	…
sent_13014	After **killing [Die]** her, he dumped her body down the same ditch where he dumped the 17-year-old girl he had previously been **convicted [Convict]** of **murdering [Die]**.
sent_13015	Tugle was on parole [Release-Parole] for this crime when he **killed [Attack]** the grandmother.
sent_13016	I understand that representing peoples is what attorneys do but even attorneys have some choice in whom they represent.

Notes: Some sentences trigger specific events.

3　Related Works

ED is one of the important tasks in NLP. Many methods have been proposed for this task. In earlier ED studies, researchers focused on employing feature-based methods[15-16], which depended on the quality of artificially designed features. Most recent works have concentrated on the representation-based neural network methods, which automatically capture the feature representations by the neural network. These methods can be roughly divided into two classes. One class is to improve ED through different learning techniques, such as CNN[2], RNN[3], GCN[11,14,17], and pre-trained models[7,12]. The other class is to enhance ED through introducing extra resources, such as document information[4-5], argument information[18], semantic information[9] and syntactic information[19-20].

Document information plays an important role in ED. Ref. [4] employed document embedding and hierarchical supervised attention mechanism to enhance event detection. Ref. [5] utilized hierarchical and bias tagging networks to model document information. The attention mechanism widely used in NLP has also been applied to ED. Ref. [18] proposed to encode argument information via supervised attention mechanisms. MOGANED[21] improved GCN with aggregative attention to model multi-order syntactic representations.

4　Conclusion

In this work, we propose a novel approach to integrate document-level and sentence-level information to enhance ED task. A hierarchical attention network is devised to automatically capture contextual information. Each event type has specific semantic information and different event types have certain semantic correlations. We deploy a shared semantic space to represent the event types and event triggers, which minimizes the distance between each event trigger and its corresponding type such that the classification of the latter is more informative and precise. Experiments on the ACE 2005 dataset verify the effectiveness of the proposed method.

References

[1] LIU J, CHEN Y B, LIU K, et al. Event detection via gated multilingual attention mechanism[C]//The 32nd AAAI

Conference on Artificial Intelligence. New Orleans, LA: AAAI Press, 2018: 4865-4872.
[2] CHEN Y B, XU L H, LIU K, et al. Event extraction via dynamic multi-pooling convolutional neural networks[C]//Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Beijing: Association for Computational Linguistics, 2015: 167-176.
[3] NGUYEN T H, CHO K, GRISHMAN R. Joint event extraction via recurrent neural networks[C]//Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Stroudsburg: Association for Computational Linguistics, 2016: 300-309.
[4] ZHAO Y, JIN X L, WANG Y Z, et al. Document embedding enhanced event detection with hierarchical and supervised attention[C]//Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). Melbourne: Association for Computational Linguistics, 2018: 414-419.
[5] CHEN Y B, YANG H, LIU K, et al. Collective event detection via a hierarchical and bias tagging networks with gated multi-level attention mechanisms[C]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP). Brussels: Association for Computational Linguistics, 2018: 1267-1276.
[6] XU G X, MENG Y T, ZHOU X K, et al. Chinese event detection based on multi-feature fusion and BiLSTM[J]. IEEE Access, 2019, 7: 134992-35004.
[7] DEVLIN J, CHANG M W, LEE K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[C]//Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis: Association for Computational Linguistics, 2019: 4171-4186.
[8] LAMPLE G, BALLESTEROS M, SUBRAMANIAN S, et al. Neural architectures for named entity recognition [C]//Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Stroudsburg: Association for Computational Linguistics, 2016: 260-270.
[9] HUANG L F, JI H, CHO K, et al. Zero-shot transfer learning for event extraction[C]//Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Melbourne: Association for Computational Linguistics, 2018: 2160-2170.
[10] SHI X, ZENG X Y, WU J, et al. Context event features and event embedding enhanced event detection[C]// Proceedings of the 3th International Conference on Algorithms, Computing and Artificial Intelligence. Sanya: Association for Computing Machinery, 2020: 1-6.
[11] NGUYEN T H, GRISHMAN R. Graph convolutional networks with argument-aware pooling for event detection[C]//The 32nd AAAI Conference on Artificial Intelligence. New Orleans, LA: AAAI Press, 2018: 5900-5907.
[12] YANG S, FENG D W, QIAO L B, et al. Exploring pre-trained language models for event extraction and generation[C]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence: Association for Computational Linguistics, 2019: 5284-5294.
[13] WANG X Z, HAN X, LIU Z Y, et al. Adversarial training for weakly supervised event detection[C]//Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis: Association for Computational Linguistics, 2019: 998-1008.
[14] CUI S Y, YU B W, LIU T W, et al. Edge-enhanced graph convolution networks for event detection with syntactic relation[C]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Punta Cana: Findings of the Association for Computational Linguistics, 2020: arXiv:2002.10757.
[15] HONG Y, ZHANG J F, MA B, et al. Using cross-entity inference to improve event extraction[C]//Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. Portland, Oregon: Association for Computational Linguistics, 2011: 1127-1136.
[16] LIU S L, LIU K, HE S Z, et al. A probabilistic soft logic based approach to exploiting latent and global information in event classification[C]//The 30th AAAI Conference on Artificial Intelligence. Phoenix: Association for Computational Linguistics, 2016: 2993-2999.
[17] LI L, JIN L, ZHANG Z Q, et al. Graph convolution over multiple latent context-aware graph structures for event detection[J]. IEEE Access, 2020, 8: 171435-171446.
[18] LIU S L, CHEN Y B, LIU K, et al. Exploiting argument information to improve event detection via supervised attention mechanisms[C]//Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vancouver: Association for Computational Linguistics, 2017: 1789-1798.
[19] SHA L, QIAN F, CHANG B B, et al. Jointly extracting event triggers and arguments by dependency-bridge RNN and tensor-based argument interaction[C]//The 32nd AAAI Conference on Artificial Intelligence. New Orleans, LA: AAAI Press, 2018: 5916-5923.
[20] LI W, CHENG D Z, HE L, et al. Joint event extraction based on hierarchical event schemas from FrameNet[J]. IEEE Access, 2019, 7: 25001-25015.
[21] YAN H R, JIN X L, MENG X B, et al. Event detection with multi-order graph convolution and aggregated attention [C]//Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). HongKong, China: Association for Computational Linguistics, 2019: 5765-5769.

编 辑 蒋 晓