Hindawi
Security and Communication Networks
Volume 2021, Article ID 5599962, 8 pages
https://doi.org/10.1155/2021/5599962

Research Article

Information Security Field Event Detection Technology Based on SAtt-LSTM

Wentao Yu, Xiaohui Huang, Qingjun Yuan, Mianzhu Yi, Sen An, and Xiang Li

PLA Strategic Support Force Information Engineering University, Zhengzhou 450000, China

Correspondence should be addressed to Mianzhu Yi; 381746262@qq.com

Received 19 January 2021; Revised 26 May 2021; Accepted 21 June 2021; Published 15 August 2021

Academic Editor: Liguo Zhang

Detecting information security events from multimodal data can help analyze the evolution of events in the security field. The Tree-LSTM network that introduces the self-attention mechanism was used to construct the sentence-vectorized representation model (SAtt-LSTM: Tree-LSTM with self-attention) and then classify the candidate event sentences through the representation results of the SAtt-LSTM model to obtain the event of the candidate event sentence types. Event detection using sentence classification methods can solve the problem of error cascade based on pipeline methods, and the problem of CNN or RNN cannot make full use of the syntactic information of candidate event sentences in methods based on joint learning. The paper treats the event detection task as a sentence classification task. In order to verify the effectiveness and superiority of the method in this paper, the DuEE data set was used for experimental verification. Experimental results show that this model has better performance than methods that use chain structure LSTM, CNN, or only Tree-LSTM.

1. Introduction

Research fields such as intelligence analysis, behavior modeling, and computational attribution models in the security field urgently need the support of a large amount of behavioral knowledge from the real world. Therefore, extracting information security behavior knowledge from massive open-source texts has become one of the current core research topics. Facing the security field, the thesis systematically studies the detection methods of behavioral knowledge and event information in massive open-source texts. By analyzing the evolution of events in the security field, the development prediction of information security events and the effectiveness evaluation of intervention actions can be realized. As an important branch of the natural language processing field, the purpose of the information extraction technology is to help users to complete information judging, comparing, and retrieving quickly and accurately. Meanwhile, as one of the key subtasks of information extraction technology, event extraction technology is becoming an important and challenging research hotspot in the field of natural language processing.

Around information extraction technology, a series of international evaluation conferences have been held, such as MUC (Message Understanding Conference), ACE (Automatic Content Extraction), and DUC (Document Understanding Conference). These conferences provide a standard test platform for the researchers who study information extraction technology and effectively improve the research level of information extraction technology. In a word, these conferences actively promoted the development of information extraction technology [1]. The event templates defined by ACE in 2005 have become a defining standard of event types generally recognized by the majority of event extraction technology researchers; these templates divide the events into 8 main types and 33 subtypes and provide corresponding training and testing corpus based on relevant templates.

There are two tasks among event extraction processes: event detection and extraction of the role of event argument. Generally speaking, almost all texts are unlabeled, such as news information in newspapers. Though these texts carry a lot of information, not every text expresses a separate event. Moreover, different event types involve different roles of

argument. To accurately and clearly extract the relevant information about the event from complex and diverse texts, event detection should be the primary task to be completed. The task of event detection is to distinguish whether a statement contains an event and which type of event it contains. The detection accuracy has a crucial impact on the subsequent identification of the role of argument. Meanwhile, the inaccurate event type is an important reason for the incorrect cascade of event extraction tasks. Currently, event trigger words have a significant impact on event detection. The definition of an event trigger is a word or phrase that can clearly express the occurrence of an event. However, the trigger word is not an indispensable basis for event detection. This paper focuses on the research of event detection tasks, with the purpose of improving the detection rate. This study has important significance for improving the event extraction technology. An efficient event extraction scheme can accurately extract events and event elements in the information security field, organize related social behaviors and behavioral premises and result knowledge, and construct an information security affair graph, which can be used in application scenarios such as causal reasoning in the information security field.

In view of these problems above, we propose a Tree-LSTM network with self-attention mechanism for the event detection model. The main research content of this paper includes the following aspects:

(1) In order to solve the restriction of the trigger word recognition performance, we treat the event detection task as a sentence classification task for processing.

(2) In the text preprocessing stage, we use existing natural language processing tools to perform word segmentation and dependency syntax analysis on text and use the GloVe word vector dictionary based on Wikipedia Chinese prediction training to vectorize the segmented data set.

(3) In order to update the hidden state results obtained by the LSTM network, we introduce a self-attention mechanism in the network training.

(4) In terms of network, we use the dependency tree structure of sentences parsed by NLP tools as the input of the tree structure LSTM network. Tree-LSTM overcomes the limitation that the linear chain structure of LSTM only allows strict sequence tasks and combines the syntactic information of the text.

The method in this paper uses syntactic information to recognize event types through event sentence classification. In the next step, researchers can use event types to assist in the joint extraction of trigger words and arguments, which not only effectively utilizes the dependence between arguments and trigger words but also can avoid error cascade.

2. Background

From the perspective of the development process, there are three main categories of event extraction technologies: method based on pattern matching, machine learning, and neural network [2].

2.1. Method Based on Feature Engineering. Before 2015, the working idea of event extraction is mainly focused on pattern matching or artificially designed features. The most prominent feature of the method based on pattern matching is that it could have good results in some specific fields, but its portability is so poor [1, 3].

2.2. Method Based on Traditional Machine Learning. The method based on traditional machine learning largely solves the poor portability of the pattern matching-based method but it is highly dependent on the labeled data. For example, Ahn divided the event detection task evaluated by ACE into 4 steps: identify event trigger words, tag event role, tag attributes, and detection output [4]. The first two steps complete the event extraction, and the rest of the steps complete the event detection task. Although these methods are more modular, easier to implement based on common components, and faster to run and debug, like other technologies based on trigger words, these methods do not pay enough attention to the multimeaning phenomenon of the text [5]. Moreover, the event detection task in these methods greatly affects the subsequent event element identification and classification effect of the event extraction system due to the error cascade problem that occurs in each step (the extraction of arguments in the pipeline method depends on the recognition result of the trigger word. If the trigger word is recognized incorrectly, it will seriously affect the recognition of the argument). Furthermore, it is time-consuming to select trigger words from sentences for labeling, and the cost of annotating the training corpus is also relatively expensive. To sum up, the recognition performance of trigger words has become a bottleneck restricting the efficiency of event detection tasks.

2.3. Method Based on Neural Network. Since 2015, some researchers have tried to use CNN/RNN for event extraction [6]. Chen et al. proposed an event extraction model called Dynamic Multipool Convolutional Neural Network (DMCNN). This model encodes each word in the sentence into a word embedding vector and adds the relative position embedding vector as a feature of auxiliary event type classification. Considering that a sentence may contain the emptying of multiple events, the model adopts a dynamic multipool approach [7]. Xu et al. proposed a Chinese event detection method based on multifeature fusion and Bi-LSTM, in which contextual information is captured based on the Bi-LSTM model. This method performs well on the CEC data set [8]. Zhang et al. proposed a new framework for event extraction based on an inverse reinforcement learning method using a generative adversarial network (GAN) and delved into the influence of two different pretraining methods of ELMo and word2vec on the proposed model [9]. Liu proposed a novel multilingual approach—dubbed as Gated Multilingual Attention (GMLATT) framework—to

simultaneously address data scarcity and monolingual ambiguity issues found in event detection [10]. To enhance both automatic feature selection and classification, Shen et al. presented an end-to-end convolutional highway neural network and extreme learning machine (CHNN-ELM) framework to detect biomedical event triggers [2]. Yu et al. built an event extraction model by using Tree-LSTM and Bi-GRU. The model obtains the vectorized representation results of candidate event sentences through Tree-LSTM and Bi-GRU, obtains the event type through sentence classification, and uses the event type as part of the event extraction input to further improve the performance and efficiency of the event extraction model [11].

3. Event Detection Model Based on SAtt-LSTM

To improve the performance of event detection, we designed an event detection model based on self-attention mechanism and Tree-LSTM. First, the model uses syntactic analysis tools to perform dependency syntactic analysis on candidate event sentences to obtain the dependency tree structure of the text. Then, we use the Chinese Wikipedia corpus to pretrain the GloVe word vector and get the word vector dictionary. Subsequently, the event sentence content document, vocabulary list, and word vector dictionary obtained after training are established for the DuEE data set to obtain the word vector representation of each word. Next, we introduce the obtained word vector into a self-attention mechanism to learn the semantic related features between discontinuous words in a sentence. The purpose of this step is to combine the obtained word vector with the semantically related features. Moreover, we combine the results above with the parsed dependency tree structure and input it into the Tree-LSTM neural network to learn sentence structure features and then obtain the vectorized representation of the candidate event sentence. Finally, we input the feature representation of the sentence into the Softmax classifier to get the final event detection result. Figure 1 is the model architecture diagram.

3.1. Text Preprocessing Module. The main purpose of dependency syntax analysis is to analyze the grammatical structure of a sentence and express it as an easy-to-understand structure. In this paper, as the input of the Tree-LSTM network, the event sentence should be constructed into a dependency tree structure firstly. In order to obtain suitable syntactic analysis tools for our model, we mainly choose two NLP tools, LTP and HanLP, to analyze the dependent syntax of sentences, respectively. Taking the sentence "Chinese public organizations have been attacked by foreign hackers using ransomware" as an example, Figure 2 shows the sentence dependency syntax tree.

Judging from the results of syntactic analysis of example sentences, the root node of the dependency tree is likely to be the trigger word in the event sentence. This proves that syntactic analysis could make good use of syntactic information, so as to carry out more accurate event element detection and event type discrimination.

When using LTP for dependency syntax analysis, we need to segment the sentence, mark the part of speech first, and then realize the dependency syntax analysis based on the results of the word segmentation and part of speech tagging. Then, according to the results of word segmentation and part-of-speech tagging, the dependency syntax analysis is realized; finally, the index and word segmentation results representing the parent node of the dependency arc are obtained as shown in Figure 3.

3.2. Sentence-Vectorized Representation Module. The dependency tree obtained after the dependency syntax analysis of the event sentence is used as the input of Tree-LSTM. Each word in the sentence is a node in the tree. Sentences are input to the Tree-LSTM network to obtain a hidden state value of each event sentence, the value is used as the output of the network, and the final judgment of the event type is based on the output [12, 13].

After getting the initial hidden state, call the self-attention mechanism model to update the hidden state and then combine the self-attention information to get the final hidden state result. The state is used for the final classification module [14].

In the process of network forward training, for each node k in the tree structure, c_k is the set of child nodes of node k and is the sum of the hidden state of the child nodes of k:

$$\widetilde{h}_k = \sum_{i \in c(j)} h_i. \tag{1}$$

The memory storage unit of Tree-LSTM is designed with a forget gate for each node. For node k, the input gate is transformed in the following way:

$$i_k = \sigma\left(W^{(i)} x_k + U^{(i)} \widetilde{h}_k + b^{(i)}\right). \tag{2}$$

The output gate is o_k switched in the following way:

$$o_k = \sigma\left(W^{(o)} x_k + U^{(o)} \widetilde{h}_k + b^{(o)}\right). \tag{3}$$

The forget door of child node i f_{ki} is switched in the following way:

$$f_{ki} = \sigma\left(W^{(f)} x_k + U^{(f)} \widetilde{h}_i + b^{(f)}\right). \tag{4}$$

Here, W and U are learnable parameters and b is the bias term.

Through this forget gate, the parent node could selectively forget the child node. Meanwhile, the input gate could be adjusted according to the semantic importance of the child node. For each node, the input gate collects all syntactic and semantic information from its child nodes, and the output gate balances meaningful information from its child nodes [15]. Finally, we designed a method to merge syntactic and semantic information into the gates of input, output and forget, and an explicit method that directly merges syntactic and semantic information into the hidden state of the node. The last step of the method is to implement a vectorized representation of the root of each tree.

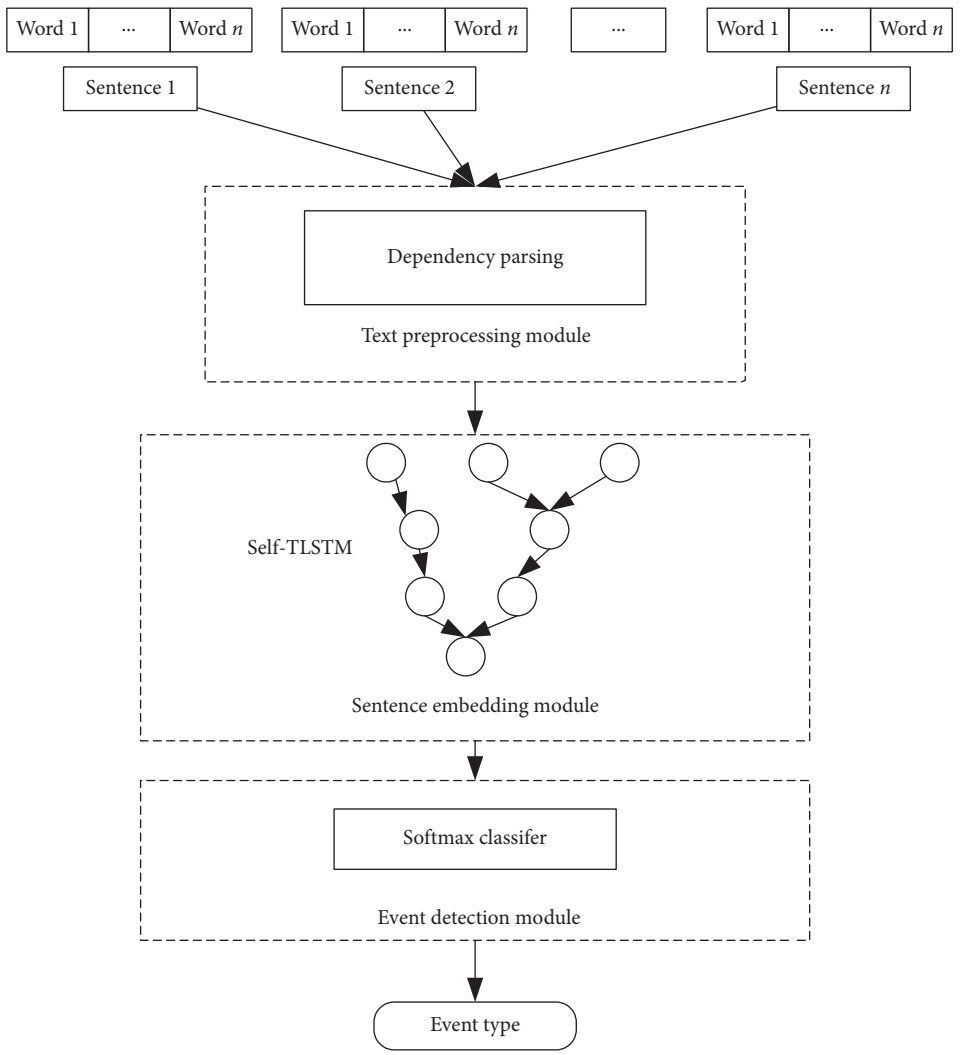

Figure 1: Model architecture.

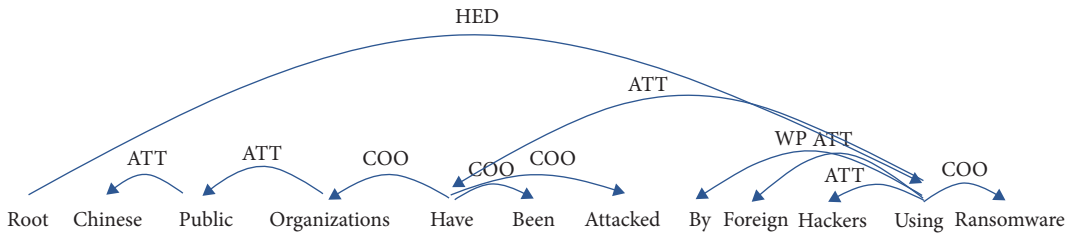

Figure 2: Display diagram of dependency syntax tree.

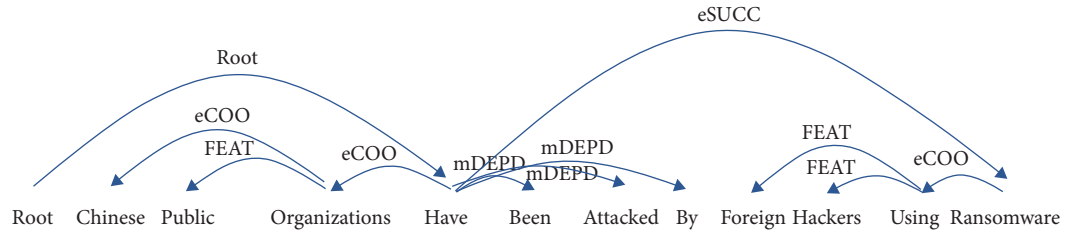

Figure 3: Example sentence processing results by LTP.

In this paper, we set the self-attention mechanism as a layer. After passing the sentence into the Tree-LSTM network, we get the initial hidden state result h_k and then the hidden state matrix: $H = [h_1, h_2, \ldots, h_n]$. In the matrix above, n is the length of a given sentence, that is, the number of words.

The idea of self-attention is to establish a connection between any two words in a sentence through a calculation step to obtain those long-distance-dependent features in the sentence. The formula is as follows:

$$g_i = \sum_{i \neq j} \alpha_{i,j} \cdot h_j. \tag{5}$$

In the formula, $\alpha_{i,j} > 0$ is the attention weight. And through Softmax regularization technology, the attention weight $\sum_j \alpha_{i,j} = 1$ is calculated as follows:

$$\alpha_{i,j} = \frac{e^{\text{score}(h_i, h_j)}}{\sum_j e^{\text{score}(h_i, h_j)}},$$
$$\text{score}(h_i, h_j) = V_\alpha^T \tan \text{h}\left(w_\alpha \left[h_i \oplus h_j\right]\right). \tag{6}$$

In the formula above, score(h_i, h_j) is achieved through MLP. MLP is used to model the correlation of words to the hidden state (h_i, h_j). The coding and decoding flow diagram of the self-attention layer is shown in Figure 4.

The self-attention mechanism is an improvement of the attention mechanism [16], which reduces the dependence on external information and is better at capturing the internal correlation of data or features. In this paper, the self-attention mechanism is used to weigh the contribution of words in determining the type of event, which effectively improves the efficiency of the model. We input the obtained hidden state matrix into the self-attention layer to update the hidden state result and use the result as the final event type classification. Figure 5 shows its structure.

3.3. Event Detection Module. In the paper, the event detection task is essentially a classification task. Its purpose is to classify each word in the sentence to see if there are elements necessary to form an event. If it is confirmed as an event sentence, then classify its event type. The classification described above is also a multiclass problem. The formula is as follows:

$$S_i = \frac{e^i}{\sum_j e^j}. \tag{7}$$

In the formula, e^i represents the i-th element in the j-dimensional array and S_i is the Softmax classification value. In multiclassification problems, the class label y could have more than two values. In the multiclassification process, Softmax maps the output of multiple neurons to the interval (0, 1). It could be understood as a probability. Based on this probability value, multiclassification is complete. The category with the largest probability in the j-dimensional array is the final classification result. It means that the element with the largest value is marked as the final classification result.

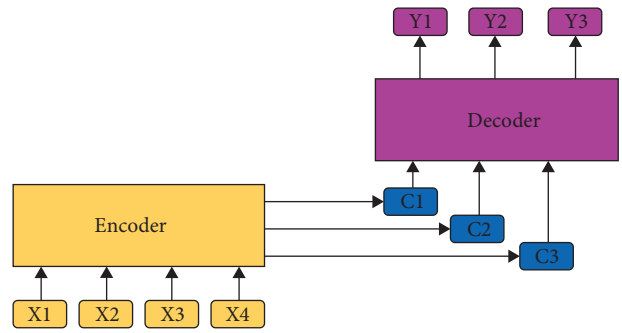

Figure 4: Structure of self-attention mechanism.

In the event detection process, after inputting the sentence vector to the Tree-LSTM network, a hidden state representation will eventually be obtained: h_k. For the hidden state of each node, Softmax classifier is used to classify the event type results. Comparing the predicted results with the actual real results, the minimized negative log-likelihood loss will be gotten. The formula is as follows:

$$\text{MLE} = \frac{1}{N} \sum_{n=1}^{n} \ln\left(1 + e^{-y_n w^T x_n}\right). \tag{8}$$

Here, x is the input value and y is the classification label. Then, the parameters of the model are optimized by the loss value.

4. Experimental Results and Analysis

4.1. Experimental Setup

4.1.1. Data Set. The data set used in the experiment is the DuEE data set provided by Baidu NLP [17]. It provides 11985 data sets that have been trained. These training data have identified the event type and marked the role of the argument. This data set divides the events into 9 types including 65 subtypes: life, judicial behavior, communication, product behavior, competition behavior, organizational relationship, finance/transaction, disaster/accident, and organizational behavior.

The experimental platform in the experiment is shown in Table 1.

4.1.2. Evaluation Index. Since the DuEE used in the paper is a Chinese data set, the pretraining of GloVe word vectors uses Wikipedia Chinese corpus. This corpus data covers a wide range and involves rich vocabulary. The paper borrows the trained Chinese Wiki Glove word vector dictionary [12]. The dictionary contains vector representations of 970,000 words, and the vector representation of each word is 100 dimensions.

4.1.3. Parameter Settings. All the experimental parameters in the experiment are shown in Table 2.

4.1.4. Evaluation Index. The paper uses precision (P), recall (R), and F1 value to evaluate the performance of the model.

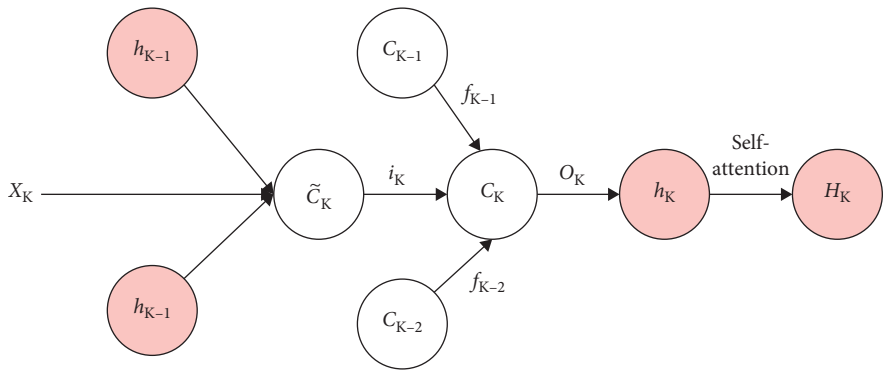

Figure 5: Structure of sentence embedding module.

Table 1: Experimental platform table.

Parameter	Value
OS	Win10
CPU	Intel Core i7-7700HQ
Memory size	16 G
Language	Python 3.6
Deep learning framework	PyTorch

Table 2: Model parameter table.

Parameter	Value
Learning rate	0.005
Batch size	25
Embedding learning rate	0.1
Weight decay	$1e-4$
Optimizer	Adagrad
Fine grain	2
Word embedding size	100
Attention dim	350

Among the formulas, TP is a positive example that is correctly identified, FP is a negative example that is classified as a positive example, and FN is a positive example that is classified as a negative example. The precision rate indicates the proportion of the samples that the model predicts to be positive, and the recall rate indicates the proportion of the positive examples that are correctly identified. The focus of the two is different, and simply considering one of them cannot reflect the performance of the model. Therefore, the paper adds the F1 value between the two as the evaluation standard. In addition, in order to evaluate the overall performance of the multiclass model, the paper adopts the overall evaluation method of macroaverage, to calculate the overall F1 value by calculating the TP, FN, and FP of all samples.

4.2. Results and Analysis. The paper selects the following methods as the baseline method to compare the performance of event detection: (1) event detection model based on pattern matching; (2) expanding the trigger vocabulary for event detection; (3) event detection model based on LSTM; (4) event detection model based on Bi-LSTM; (5) event detection model based on Tree-LSTM.

4.2.1. Experimental Results. It could be seen from Table 3 that the Tree-LSTM event detection model based on the self-attention mechanism has achieved excellent results in all aspects compared to other methods. In terms of accuracy index, the model in this paper is slightly lower than the event detection model using Bi-LSTM network. Compared with other models, it shows a significant improvement. Meanwhile, the recall rate and F1 have also been significantly improved.

4.2.2. Experiment Analysis. Analyzing the experimental results, we can see that this model is superior to other models for the following reasons:

(1) Model Selection. Compared with the previous RNN model, LSTM adds the concept of forgetting gate. Through the training process, the model could learn what information should be remembered and what information should be forgotten and could do well to capture longer-distance dependencies. Therefore, compared with previous methods based on pattern matching and expanded trigger vocabulary, LSTM has a significant improvement in effect and its portability is better. Then, the Bi-LSTM-based method is used to capture sentence dependencies in two directions through the network, which improves the understanding of text semantics. Therefore, compared with LSTM, the accuracy rate is increased by nearly 5 percentage points, and the F1 value is also increased by about 8 percentage points. Compared with Bi-LSTM, Tree-LSTM uses a tree topology, and its input of the network is the sentence processed by the NLP tool. Through this approach, it could better combine the syntactic information of sentences and avoid the problem that the dependency between long-distance word pairs is difficult to obtain. Compared with the Bi-LSTM-based method, the recall rate is increased by 6.4 percentage points. In this paper, we select a SAtt-TLSTM-based model. Based on Tree-LSTM, we introduce a method that could directly calculate the dependency relationship regardless of the distance between words to learn the internal structure of a sentence. To update the initial hidden state of the network output, we implement a relatively simple parallel computing self-attention mechanism. Compared with Tree-LSTM, the recall rate of this model has increased by nearly 7 percentage

Table 3: Baseline methods and results.

Baselines	P	R	F1
Pattern matching	0.627	0.508	0.561
Expanded trigger vocabulary	0.571	0.642	0.697
LSTM	0.676	0.655	0.626
Bi-LSTM	0.720	0.680	0.699
Tree-LSTM	0.634	0.744	0.709
SAtt-LSTM	0.691	0.812	0.731

points, and the F1 value has also increased by 2.2 percentage points. It should be the best model among all baseline methods.

(2) Application of Self-Attention Mechanism. The paper introduces a self-attention mechanism that could learn the correlation between the current word and the previous part of the sentence. It could judge the importance of the current word according to the strength of the correlation. To a certain extent, it solves the problem of the inability to highlight keywords. The reason for the problem is that the model in the paper does not remove the stop words. In other words, the critical words should be the trigger words in an event. After combining the self-attention mechanism with the original dependency tree result, the trigger word becomes more prominent. The above thesis has great advantages in the discrimination of event types. The results show that the model combined with the self-attention mechanism performs better than the Tree-LSTM model without the self-attention mechanism.

4.2.3. Model Performance Factors

(1) Word Segmentation Mechanism. The result of word segmentation is an important element that affects the experiment process and results. Through experimental comparison, we found that using different tools to do word segmentation and syntactic analysis of the same text finally got different indicators. As shown in Table 4 and Figure 6, the final result obtained by using LTP is slightly better than that of HanLP. However, in the training process, the text using LTP word segmentation and dependent syntax analysis requires longer time for training, and it also needs more rounds to reach the optimal parameters. In fact, whether it is LTP or HanLP used in the paper or other systems such as Stanford NLP or jieba word segmentation, its ability to recognize unregistered words needs to be improved, and an external dictionary is needed to expand its thesaurus. These defects double the workload of preprocessing. Furthermore, it is basically impossible to build a dictionary containing all knowledge bases. At present, the Chinese classification of "words" has not formed an officially recognized unified standard. This is also one of the key reasons for the difficulty of Chinese word segmentation. Different segmentation results are particularly important for accurately obtaining its word vector. For those words with special symbols, if the symbols and words cannot be separated well during word segmentation, it will lead to the inability to find the corresponding word vector in the word vector, which will affect the final experimental results.

Table 4: Word segmentation methods and results.

Word segmentation method	P	R	F1
LTP	0.698	0.812	0.734
HanLP	0.691	0.812	0.731

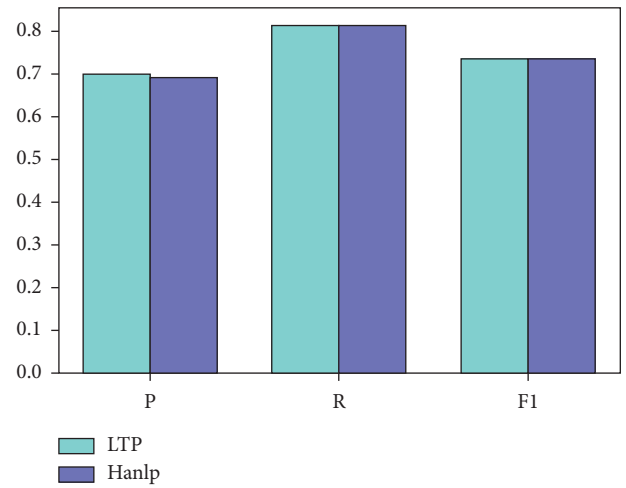

Figure 6: The effect of different word segmentation methods.

(2) Removing Stop Words. In this model, after the dependent syntactic analysis of the event sentence, the word segmentation result is directly obtained from the syntactic analysis result. Then, in order to integrate the embedded sentence vector and the size of the dependency tree, the model does not remove stop words from the word segmentation results. In the LSTM model, in order to test whether the removal of stop words has an effect on the final result, we conducted comparative experiments, respectively, and the results proved that better experimental results could be obtained by removing the stop words. Stop words are mostly auxiliary words of connecting sentences, which have little effect on the actual content of the text. Therefore, after removing the stop words in the sentence, the core elements of the sentence could be better analyzed, and then the better analysis results could be obtained.

5. Conclusions

The paper focuses on the critical first step in the event extraction task. The result of the event detection task could directly affect the subsequent extraction of the entire event. Aiming at the shortcomings of previous methods, we treat the event detection task as a sentence classification task. By using the Tree-LSTM network that introduces a self-attention mechanism, we propose an event detection model. Firstly, the model uses the GloVe word vector to express the text vocabulary, starting from the bottom, focusing on the use of global information in the text. Next, the dependency tree of candidate event sentences is used as the input of the model to solve the shortcomings of insufficient use of

syntactic information in existing event detection models. Then, we use the Tree-LSTM network which introduces the self-attention mechanism to obtain the vectorized representation of the candidate event sentence. Finally, the task of event detection was completed by using Softmax. After training and testing on the DuEE dataset, the results obtained prove that the model proposed in the paper is superior to the previous model. It not only improves the accuracy of event detection but also provides good basic information for the role classification task of event extraction. This model fundamentally optimizes the problem of error cascade caused by incorrect trigger word recognition in the task of event extraction. At the same time, it further proves the feasibility and advantages of deep learning in event detection and event extraction technology.

Data Availability

The Baidu previously reported DuEE1.0 data were used to support this study and are available at https://aistudio.baidu.com/aistudio/competition/detail/65. The datasets are cited at relevant places within the text as references.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] T. H. Nguyen, K. Cho, and R. Grishman, "Joint event extraction via recurrent neural networks," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 300–309, HLT, San Diego, CA, USA, June 2016.

[2] C. Shen, H. Lin, X. Fan et al., "Biomedical event trigger detection with convolutional highway neural network and extreme learning machine," *Applied Soft Computing*, vol. 84, 2019.

[3] B. L. Hu, R. F. He, H. Sun, and W. J. Wang, "Chinese event type recognition based on conditional random fields," *Progress in Artificial Intelligence*, vol. 25, no. 3, pp. 445–449, 2012.

[4] D. Ahn, "The stages of event extraction," in *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pp. 1–8, Sydney, Australia, July 2006.

[5] K. Zhang, Y. Guo, X. Wang, J. Yuan, Z. Ma, and Z. Zhao, "Channel-wise and feature-points reweights DenseNet for image classification," in *Proceedings of IEEE International Conference on Image Processing (ICIP)*, Taipei, Taiwan, September 2019.

[6] X. Fan, H. Lin, Y. Diao, and Y. Zou, "An integrated biomedical event trigger identification approach with a neural network and weighted extreme learning machine," *IEEE Access*, vol. 7, pp. 83713–83720, 2019.

[7] Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao, "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*, vol. 1, pp. 167–176, Beijing, China, July 2015.

[8] G. Xu, Y. Meng, X. Zhou, Z. Yu, X. Wu, and L. Zhang, "Chinese event detection based on multi-feature fusion and BiLSTM," *IEEE Access*, vol. 7, pp. 134992–135004, 2019.

[9] T. Zhang, H. Ji, and A. Sil, "Joint entity and event extraction with generative adversarial imitation learning," *Data Intelligence*, vol. 1, no. 2, pp. 99–120, 2019.

[10] J. Liu, Y. Chen, K. Liu, and J. Zhao, "Event detection via gated multilingual attention mechanism," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, February 2018.

[11] W. Yu, M. Yi, X. Huang, X. Yi, and Q. Yuan, "Make it directly: event extraction based on tree-LSTM and Bi-gru," *IEEE Access*, vol. 8, pp. 14344–14354, 2020.

[12] W. Liu, P. Liu, Y. Yang, Y. Gao, and J. Yi, "An attention-based syntax-tree and tree-lstm model for sentence summarization," *International Journal of Performability Engineering*, vol. 13, no. 5, p. 775, 2017.

[13] M. Ahmed, M. Rifayat Samee, and R. E. Mercer, "Improving tree-LSTM with tree attention," 2019, https://arxiv.org/abs/1901.00066.

[14] L. Mou, G. Li, L. Zhang, T. Wang, and Z. Jin, "Convolutional neural networks over tree structures for programming language processing," 2016, https://arxiv.org/abs/1409.5718.

[15] S. MacAvaney, L. Soldaini, A. Cohan, and N. Goharian, "GU IRLAB at SemEval-2018 task 7: tree-LSTMs for scientific relation classification," 2018, https://arxiv.org/abs/1804.05408.

[16] K. Zhang, N. Liu, X. Yuan et al., "Fine-grained age estimation in the wild with attention LSTM networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 9, pp. 3140–3152, 2020.

[17] X. Li, F. Li, L. Pan et al., "DuEE: a large-scale dataset for Chinese event extraction in real-world scenarios," in *Proceedings of the International Conference on Natural Language Processing and Chinese Computing*, pp. 534–545, Springer, Zhengzhou, China, October 2020.