Exploring Multiple Embedded Features on Event Extraction

Shi-Xiang Yi* and Chun-Yan Li
Key Lab of the Ministry of Education, School of Automation, Chongqing University, Chongqing, 400044, China

*Corresponding authors: yishixiang@gmail.com

Abstract. In recent years, the neural network method can automatically learn effectively features. Unlike traditional discrete features, neural network features are mostly continuous features and can be automatically combined to build higher-level features. The efficiency of the features has been proven in numerous tasks in natural language processing and has led to breakthroughs. In this paper, we propose a event extraction system based on combination of multiple embedded features. Our work is mainly based on the three aspects: (1) traditional pipeline systems have serious error propagation problems; (2) there are several different event descriptions in the text; (3) representation learning can provide rich semantic and syntactic representation. As a result, we achieve competitive performance, specifically, F1-measure of 60.25 in event extraction. Meanwhile, evaluation results point out some shortcomings that need to be addressed in future work.

1. Introduction
In such an era of information overload, how to obtain valuable information that people want from massively fragmented text information is a huge challenge. Moreover, a lot of inflammatory titles make readers more suspicious towards news content. Information extraction technology can effectively solve this problem. The general goal of information extraction is to extract specific structured information for text. Therefore, this method can automatically mine key information from massive news data and help to further classify, extract, fuse and verify information. Research in this field is based primarily on Automatic Content Extraction (ACE). The ACE defines the information to be extracted for the event extraction task:

- Event Mention: A phrase or sentence describe the event, containing a trigger and any number of arguments.
- Event Trigger: The word in the event description that best represents the occurrence of the event, and it is generally a verb or a noun.
- Event Argument: Important information of an event consists of fine-grained units that express complete semantics, such as entities and attribute values.
- Argument Role: The role of the event element in the event, the semantic relationship between the event element and the event.

A number of event extraction frameworks are feature-based approaches to apply to sentence-level or document-level. Feature-based methods mainly rely on a large number of feature engineering for machine learning models, from local features to high-level feature representations. Gupta [1] chose to find the cross-document anchoring features to seek the event extraction model applicable to different domains. Liao [2] tagged the easier cases, and use such knowledge to help tag the harder cases.
In recently, state-of-the-art event extraction system uses neural network models to improve event extraction. Neural networks have recently been introduced into event extraction with the goal of overcoming the two basic limitations of traditional feature-based methods: (1) feature engineering for rich feature sets; (2) error propagation caused by feature extraction phase. All of these models use word embedding, a generic word representation generated through unsupervised training on deep learning models. Jagannatha and Yu [3] use bi-directional recurrent neural network to extract instances of events in health records. Feng [4] developed a hybrid neural network (CNN and RNN) that captures sequence information and block information from sentences, enabling training of an event detector that does not rely on any manual features.

However, to our knowledge, there is no comprehensive research on concatenating embedded features in a pipeline model. The remainder of this paper is organized as follows: We present the motivation for building models by evaluating existing models (section 2). Subsequently, we explain the model proposed in this paper in detail (section 3), including the trigger classification model and the argument role classification model. Finally, we conduct experiments based on the model proposed in this paper (section 4) and conclude (section 5).

2. Motivations
In this section we analyze the event extraction method mentioned above and present our solutions based on error analysis for the baseline event extraction.

2.1. Error Propagation
The traditional event extraction method divides the whole process into event recognition task and argument role classification task. And there are multiple subtasks in each task, such as trigger recognition, event classification, etc. If the previous task is biased, subsequent work will no longer be feasible and effective. Björne's research shows that more than 60% of the errors in event extraction are derived from trigger recognition tasks. Therefore, the completion of the trigger word recognition with high quality is a guarantee for subsequent public safety event extraction.

In this work, we use the representation learning method of embedded features to solve this problem. Embedding essentially casts features into a higher-dimensional space based on the tasks. This allows the classifier to learn representation better and more comprehensively. Embedding means converting data into feature representations, some of which can be represented by the distance of the vector distance.

2.2. Feature Engineering
Feature-based event extraction systems require natural language processing toolkits to extract various types of local features and global features. We summarize the commonly used features below.

(1) Local features of argument: context of the argument mention, type of argument, type of event, common ancestor node between arguments in the parse tree, relative position of current argument and trigger and whether there are candidate arguments of the same type.
(2) Local features of trigger: type of trigger, distance between current argument and trigger, syntactic dependency type with trigger words, the closest entity type with the trigger and the path from the leaf node of the trigger to the root in the parse tree.
(3) Cross-domain features: entity consistency, trigger consistency, co-occurrence with trigger in clause and entity-subtypes of arguments.

The above features rely on traditional external NLP tools such as dependency analysis, syntax analysis, and part-of-speech tagging in the feature extraction process. Therefore, it would cause error accumulation. In addition, some languages and fields do not have such processing tools, and the features need to be manually set. A well-designed manual feature can bring significant improvements, but in contrast, it can make the system slow and difficult to handle in large-scale applications.

In order to solve this problem, we choose Bi-LSTM (Bi-directional Long Short Term Memory) and CNN (Convolutional Neural Network) to adaptively extract text features. Deep model can perform well on these types of tasks without any complicated and time consuming feature engineering. Most of
the time, these features require domain knowledge, creativity, and a lot of trial and error. A more detailed model structure is described in Section 3.

2.3. Multiple events Per Sentence
Event extraction methods mainly include the extraction method based on trigger words and the extraction method based on event instances. The detection method based on event instances is to use sentences instead of words as identification instances. Then, it is transformed into a sentence clustering problem by clustering method, and event sentences are obtained by clustering. So this method could only be used to extract text with an event and cannot solve the event extraction problem of long text in practice. Consider the following sentence,

In Baghdad, a cameraman died when an American tank fired on the Palestine Hotel.

There are two events in the above sentence, include Life.Die and Conflict.Attack. Since a sentence may contain multiple events, the event extraction method based on trigger words is more accurate and comprehensive for event extraction. In this paper, we regard semantic role labeling as a classification task in which candidate arguments forms a phrase with the trigger word. Then, we concatenate the embedding feature from the proposed trigger classification model with other embedding features. Our model is different from the traditional pipeline model, and it can fully utilize the event type information represented by the trigger word embedding.

3. System Approach Overview

3.1. Event trigger classification
The Bi-LSTM based event trigger classification method proposed in this study is shown in figure 1. The system can be divided into three parts: (1) The input part completes the splicing of the word embedding vector and the remaining word features; (2) The high-level semantic representation acquisition part is completed by the Bi-LSTM network model; (3) the forward neural network model to complete the final trigger word classification task, and its input combine the word embedding and the semantic features of the text. Then we will briefly describe the various parts of the model.

For each word \( w_i \in W \) in the text, we concatenate two features. \( E_w \in \mathbb{R}^{w \times d} \) represents pre-trained word embedding, and \( E_o \in \mathbb{R}^{o \times d} \) represents other features of a word (entity type, part of speech type, and syntactic type). \( n_w, n_o \) respectively represent the length of the vocabulary and the number of other feature types. \( d_w, d_o \) respectively represent the dimension of the word embedding and the dimension of the embedded features. Each word in the text is mapped to a dense vector representation by two word-feature embedded vector lookup tables.

3.2. Argument classification
We models the argument prediction task as a relation extraction problem, pairs the trigger and the argument, and classifies the relationship. We use convolutional neural networks to classify arguments by concatenating multiple embedded features (word embedding, position embedding, trigger word embedding) that is shown in figure 2. Given a piece of text \( T \) with a length of \( l \), each candidate argument \( w_{Arg} \in T \) can construct a contextual word pair \((w_{Arg}, w_{Tri})\) with the trigger word \( w_{Tri} \) in text.

The main parts of the model are explained below.

(1) Word embedding table: construct word embedding with pre-trained word embedding \( word \), and the length is \( m_w \). (2) Relative position embedding table: the text can be represented as \( T = [w_1, w_2, w_{Arg}, \ldots, w_{j}, \ldots, w_l] \). The relative position mainly measures the distance between the word and the candidate argument and the distance between the word and the trigger word. Then the relative distance ranges from \(-l+1\) to \(l-1\), and the length of position embedding \( p_i \) is \( m_p \). (3) Trigger word
embedding table: trigger word embedding $Tri$ is determined by the input of forward neural network in the previous event trigger classification model.

The feature vectors $x_i = \left[ \text{word}_i, p^{(\text{argu})}_i, p^{(\text{tri})}_i, Tri_i \right]$ of each word in the text are obtained from the three types of features mentioned above. Thus, the whole text could be represented by $X$ with size of $(m_w + 2m_p + m_t) \times l$. The matrix $X$ is used as an input to perform a convolution operation with the convolution kernel. After that, we choose max-pooling layer to retain the strongest feature of the feature vectors obtained after each convolution, and concatenate these features to get the final high-level feature representation. In the end, softmax function predicts the type of argument. The selection of the max-pooling layer is based on the following considerations: The operation of max-pooling ensures the invariance of position and rotation, condenses variable-length feature map into a representation of fixed length, and effectively reduces the number of hyper-parameter of the model.

![Figure 1. Trigger word classification model](image1)

![Figure 2. Argument classification model](image2)

4. Experimental Results

The experimental environment of this paper is as follows: the operating system is Ubuntu-16.04, the CPU is Intel Core i7-8700K, the GPU is GeForce GTX-1070Ti, and the development environment is Tensorflow-1.10. We adopt the CEC (Chinese emergency corpus) built by Shanghai University. It contains five social news reports of earthquakes, fires, traffic accidents, terrorist attacks and food poisoning. Similar to the dataset provided by ACE2005, CEC marks the six most important event extraction information in the corpus: Event, Denoter, Time, Location, Participant, and Object.

4.1. Event trigger classification

We use training set and development set to select the hyper-parameter of the model. The forward neural network has two hidden layers and the number of neurons in the hidden layers is 130 and 110 respectively. Cross entropy is selected as the loss function and the probability of dropout is set to 0.3. Furthermore, the number of LSTM layers is 2 and the number of cells in the hidden layer is 200. We choose the pre-training model of Bert [5] and integrate it into the task model.

In table 1, the model proposed in this paper has a certain improvement under the F1-score compared with the previous benchmark model. And the performance of the support vector machine model is obviously inferior to our model. It proves that the heavy dependence on the NLP tool would continuously introduce errors, and the features obtained by the empirically determined feature engineering cannot effectively represent the text information. CNN-based trigger classification method significantly reduces recall rate due to fixed input dimension in CNN model. It means that the convoluted object must have local correlation, and CNN-based model cannot learn long dependencies.
Table 1. Comparison of trigger classification model

| Model                        | Precision | Recall | F1-score |
|------------------------------|-----------|--------|----------|
| Our model                    | 78.68     | 77.93  | 78.30    |
| CNN [6]                      | 79.21     | 76.53  | 77.85    |
| SVM [7]                      | 76.45     | 71.04  | 73.65    |
| SVM+Embedding [8]            | 77.85     | 73.62  | 75.68    |

In table 2, we also focus on the impact of different features on model performance. $E_w$, $E_o$ respectively represent word embedding and other embedding features (lexical, syntactic, entity). It can be seen that the syntactic features are the most helpful for the event trigger word classification task and the part of speech information is usually used as an auxiliary feature of tasks such as semantic role labeling and entity recognition. Furthermore, the entity type can represent the components of the event to a certain extent, such as time, place and object, thus it helps the trigger classification task more.

Table 2. Comparison of features

| No. | Features                        | F1-score |
|-----|---------------------------------|----------|
| 1   | $E_w, g$                        | 76.61    |
| 2   | $E_w, g$ (part of speech tag)   | 77.59    |
| 3   | $E_w, g$ (entity type)          | 78.81    |
| 4   | $E_w, g$ (dependence relationship) | 78.47   |

4.2. Argument classification

In this section, we compare the model proposed in this paper with other models of classical work. What's more, we compare the model effects after not combine such features to measure the validity of the event trigger embedding and the position distance embedding. The size of the convolution kernels in the convolution layer are $\{3,4,5\}$, and there are 400 convolution kernels of each size. The Position embedding set to 50 dimensions and word embedding set to 100 dimensions. The training optimization method uses the Adam algorithm with a learning rate of $10^{-3}$.

The results are shown in table 3. Accuracy rate of across documents model and feature-based event extraction model are quite competitive owing to combine multiple features (Syntactic and semantic). But in comparison the model proposed in this paper only relies on automatic feature extraction, which has low dependence on various natural language processing tools. Simultaneously, our model has the highest recall rate, because it not only utilizes the semantic information of the sentence, but also uses the event type information introduced by the trigger word embedding feature, which can effectively guide the classification of argument characters. It should be noted that information about event types is also used in the model proposed in [9]. However, since its feature is discrete, the representation of event type information is still not as rich as that of dense embedding.

Table 3. Comparison of arguments classification model

| Model                                      | Precision | Recall | F1-score |
|--------------------------------------------|-----------|--------|----------|
| Our model                                  | 63.14     | 57.62  | 60.25    |
| Cross-event [2]                            | 45.17     | 44.12  | 44.64    |
| Cross-entity [9]                           | 51.65     | 45.58  | 48.43    |
| Joint learn with local and global features [10] | 62.72     | 48.55  | 54.73    |
| Dynamic multi pooling CNN [11]             | 61.28     | 45.86  | 52.46    |
In order to analyze the importance of embedded features, we set the base-line Model, in which only word embedding and position embedding of candidate argument are used. Then we add the trigger embedding feature and the trigger word distance feature in turn and compare the performance of the model. The result is shown in table 4. Trigger embedding has the biggest effect on the model. This is because the trigger embedding can effectively represent the type information of the event, which is of great help to the classification of argument characters.

**Table 4. Comparison of the importance of features**

| Model                     | Precision | Recall | F1-score |
|---------------------------|-----------|--------|----------|
| Our model                 | 63.14     | 57.62  | 60.25    |
| + Trigger embedding       | 59.38     | 51.45  | 55.13    |
| + Trigger position embedding | 34.22     | 21.61  | 26.49    |
| Base-line model           | 28.74     | 21.53  | 24.62    |

5. Conclusion

In this work, we classify the argument role as a classification task that forms a phrase with the trigger word. In order to get rid of the dependence on feature engineering, the model concatenates word embedding, trigger embedding and distance embedding to construct the representation of text. As a result, we achieve competitive performance, specifically, F1-measure of 60.25 in event extraction. The experimental results also show that the existing trigger classification model does not reach the ideal state, so the accuracy of the event extraction model cannot be further improved. Our future work will take advantage of the nonlinear combination of neural networks and try to build a joint model.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant Nos. 61773080) and the Basic Science and Research Technology Major Project of Chongqing (Grant Nos. cstc2015jcyjB0569).

References

[1] Gupta R and Sarawagi S 2009 Domain adaptation of information extraction models ACM SIGMOD Record 37(4): 35-40
[2] Liao S and Grishman R 2010 Using document level cross-event inference to improve event extraction Proceedings of the 48th ACL 789-797
[3] Jagannatha A N and Yu H 2016 Bidirectional RNN for medical event detection in electronic health records Proceedings of the conference. Association for NAACL Chapter 473
[4] Feng X, Qin B, and Liu T 2018 A language-independent neural network for event detection Science China Information Sciences 61(9): 092106
[5] Devlin J, Chang M W, Lee K, et al. 2018 Bert: Pre-training of deep bidirectional transformers for language understanding (Preprint arXiv:1810.04805)
[6] Wang J, Li H, An Y, et al. 2016 Biomedical event trigger detection based on convolutional neural network International Journal of Data Mining and Bioinformatics 15(3): 195-213
[7] Pyysalo S, Ohta T, Miwa M, et al. 2012 Event extraction across multiple levels of biological organization Bioinformatics 28(18): i575-i581
[8] Zhou D, Zhong D, He Y 2014 Event trigger identification for biomedical events extraction using domain knowledge Bioinformatics 30(11): 1587-1594
[9] Hong Y, Zhang J, Ma B, et al. 2011 Using cross-entity inference to improve event extraction Proceedings of the 49th ACL: Human Language Technologies 1: 1127-1136
[10] Li Q, Ji H, and Huang L 2013 Joint event extraction via structured prediction with global features Proceedings of the 51st ACL 1: 73-82
[11] Chen Y, Xu L, Liu K, et al. 2015 Event extraction via dynamic multi-pooling convolutional neural networks Proceedings of the 53rd ACL 1: 167-176