Comput Syst Sci & Eng (2020) 5: 311–319


International Journal of
Computer Systems
Science & Engineering

# Event Trigger Recognition Based on Positive and Negative Weight Computing and its Application

**Tao Liao[1]*, Weicheng Fu[1]†, Shunxiang Zhang[1]‡ and Zongtian Liu[2]§**

[1] *School of Computer Science and Engineering, Anhui University of Science and Technology, Huainan 232001, China*
[2] *School of Computer Engineering and Science, Shanghai University, Shanghai 200072, China*

Event trigger recognition is a sub-task of event extraction, which is important for text classification, topic tracking and so on. In order to improve the effectiveness of using word features as a benchmark, a new event trigger recognition method based on positive and negative weight computing is proposed. Firstly, the associated word feature, the part-of-speech feature and the dependency feature are combined. Then, the combination of these three features with positive and negative weight computing is used to identify triggers. Finally, the text classification is carried out based on the event triggers. Findings from our experiments show that the application of our method achieves ideal results.

Keywords: Event trigger recognition, Associated word, Dependency, Part-of-speech, Positive and negative weight computing, Text classification

## 1. INTRODUCTION

With the rapid development of the Internet and mobile Internet, the volume of data is increasing rapidly, making big data a hot research topic. At the same time, various emergency situations occur frequently, often reported on the Internet. Hence, the research on events in the big data environment is attracting a great deal of attention.

An event refers to something that occurs at a specific time and in a particular environment, involves several actors, and exhibits several behavioral characteristics (Liu et al. 2009). The events in texts have important implications for the fields of automatic summarization, text classification, topic tracking, and information retrieval and so on. However, a computer cannot directly recognize the events in the texts that will affect subsequent applications, so it is especially important to automatically extract such events. Event extraction (EE) is a sub-task of information extraction whereby event information is extracted from unstructured text in a structured form for subsequent use. Event extraction generally comprises two steps: event detection (ED) and event argument extraction (AE). The focus of this paper is on event trigger recognition, which is the main task of ED. Event triggers, which can clearly indicate the occurrence of an event, are mostly verbs and nouns. The purpose of event trigger recognition is to find event triggers in texts such as news texts and social media texts, which have a great impact on the effect of EE.

In the event trigger recognition process, we can use the word feature to identify the event trigger. This method takes the event trigger that appears in the training set as a feature, and when it also appears in the test set, we identify it as an event trigger. Although this method is simple and the recall rate is high, it has poor precision, resulting in poor final event recognition. However, we can use the word feature to identify the event trigger as a benchmark and add other features to improve event trigger recognition. Nevertheless, there are two challenges:

1) There are few features that greatly improve the result of the benchmark application.

*tliao@aust.edu.cn
†1425548176@qq.com
‡Email of corresponding author: sxzhang@aust.edu.cn
§ztliu@shu.edu.cn

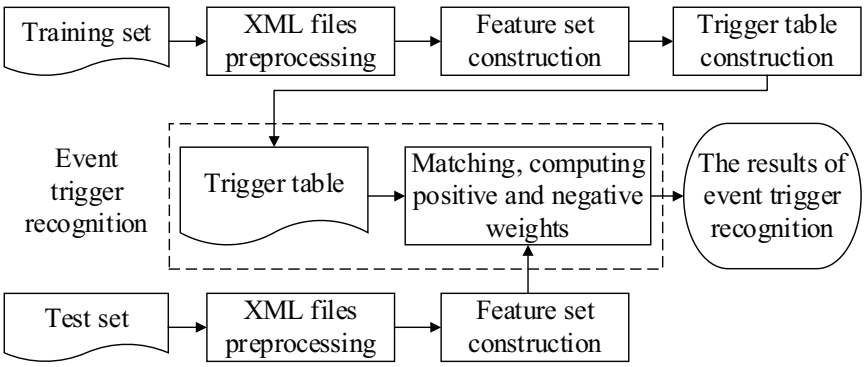


**Figure 1** Architecture of event trigger recognition system based on positive and negative weight computing.

2) The result achieved by using other features to improve event trigger recognition is still not ideal.

In order to solve these two problems, we first define a feature called the 'associated word', which can greatly improve the result of benchmark. After that, the single feature will be categorised as either a positive or negative feature which can be used for weight computing. The positive and negative weight computing method is used to identify the event triggers, and the result of the benchmark combined with a single feature is improved. Finally, based on the benchmark, we combine multiple features with positive and negative weight computing, and obtain the final and best event trigger recognition result.

The architecture of the event trigger recognition system in this paper comprises four parts: XML files preprocessing, feature set construction, trigger table construction and event trigger recognition, as shown in Figure 1. First, XML files preprocessing and feature set construction are conducted for the entire corpus. Then, by processing the training set of the corpus, the trigger table is constructed. Finally, the candidate triggers and their feature sets in the test set are matched in the trigger table, and the positive and negative weight computing is performed to complete the event trigger recognition process.

The rest of this paper is organized as follows: Section 2 describes related works. Section 3 introduces an event trigger recognition method based on positive and negative weight computing and the corresponding experiments. Section 4 proposes a text classification method based on event triggers and the experiment results. Conclusions are drawn in Section 5.

## 2. RELATED WORKS

Existing event trigger recognition methods mainly include rule-based methods and machine learning methods.

### 2.1 Rule-Based Methods

In the field of rule-based methods, Zhao et al. (2008) expanded triggers by using synonyms, enlarging the scale of triggers, and effectively mitigating the imbalance between positive and negative data. Alfred et al. (2018) proposed a knowledge acquisition framework that uses domain-specific ontology to encode events for later search and analysis, which has higher accuracy and lower cost. Fabio et al. (2018) proposed a new cross-media event extraction and event co-reference system that identifies a link event extracted from two complementary data sources by a novel co-reference mechanism. Yang et al. (2019) were the first to propose an event extraction model to overcome the role overlap problem by separating parameter predictions according to roles. Cao et al. (2018) combined the ACE (Automatic Context Extraction) event pattern and the TABARI (Textual Analysis by Augmented Replacement Instructions) event pattern to generate new patterns. Li et al. (2019) proposed a joint event extraction method that extracts frames representing event information from FrameNet. Huang et al. (2016) proposed a new liberal event extraction paradigm, which combines symbol and distribution semantics to detect and represent event structures. Laparra et al. (2017) suggested a method for extracting timelines, participants, locations, and times of events from multi-language and cross-language data sources. Liu et al. (2017) propose a non-entity parameter extraction method based on structure representation. It directly uses the syntactic relationship between the event trigger and the syntactic subtree to identify non-entity parameters. Li et al. (2019) demonstrated a comprehensive multilingual knowledge extraction, aggregation, and visualization system.

### 2.2 Machine Learning Methods

In the field of machine learning, Ahn (2006) achieved good results by combining the nearest neighbor classifier with the maximum entropy classifier for event trigger recognition. Nguyen and Grishman (2015) used CNN (Convolutional Neural Networks) to recognize event triggers. Liu et al. (2017) improved the F1 measure by using a supervised attention mechanism combined with argument information for event trigger recognition. Liu et al. (2018) utilized consistency information in multilingual data and complementary information in multilingual data transmission through the context attention mechanism. Feng et al. (2018) proposed a language-independent neural network to capture sequence and block information for a particular context and use them to train a multilingual event detector that does not require any manual coding features. Li et al. (2019) created a new knowledge-driven tree-structured LSTM (Long Short-Term Memory) networks framework. It has achieved good results in biomedical event extraction. Ghaeini et al. (2016) proposed a

**Table 1** XML file fragment preprocessing example.

| Preprocessing methods | Results |
|---|---|
| XML file fragment | <Event eid="e6"><br><Time tid="t6" type="relTime">目前</Time>已造成<br><Participant sid="s6">8人</Participant><br><Denoter type="stateChange" did="d6">死亡</Denoter>。<br></Event> |
| Remove annotation | "目前已造成 8 人死亡。" |
| Segmentation | ["目前", "已", "造成", "8", "人", "死亡", "。"] |
| Trigger index | [6] |
| Part-of-speech tagging | [“nt”, “d”, “v”, “m”, “n”, “v”, “wp”] |
| Dependency parsing | {3: ADV, 3: ADV, 0: HED, 5: ATT, 6: SBV, 3: VOB, 3: WP} |

language-independent event detection method based on RNNs (Recurrent Neural Networks), which can automatically extract the effective features of the original text to detect valuable events. Zhang et al. (2015) introduced a new unsupervised algorithm that uses a novel probabilistic graphical model to cluster sentences that describe similar events in parallel news streams. Nguyen and Grishman (2018) studied a convolutional neural network based on dependency tree to perform event detection and proposed a new pooling method based on entity mentions to aggregate convolution vectors. Zhang et al. (2017) proposed a multimodal event extraction algorithm based on text feature and visual mode joint training event extractor. Nagesh et al. (2017) were the first to identify a message containing an event by using a binary classifier. They then used a CRF (Conditional Random Field) to extract events from the message and used a parts-of-speech approach to resolve some of the noisy problems associated with social media text. Ferguson et al. (2018) proposed a self-training event extraction method that derives additional training data by using multiple mentions of the same event instance from news source articles from multiple sources. Huang et al. (2018) proposed a transferable nervous system architecture. This architecture improves the scalability of event extraction while saving manual work. Abulaish et al. (2019) suggested a multi-attribute graph-based approach for text data modeling and event detection in Twitter.

From the above researches at home and abroad, it can be seen that, whether they are rule-based or machine learning-based event trigger recognition methods, most of them need to obtain features and then carry out subsequent research work. Therefore, we consider adding different features to assist event trigger recognition based on word features. At the same time, the positive and negative weight computing method is introduced to further improve the performance on event trigger recognition.

# 3. EVENT TRIGGER RECOGNITION BASED ON POSITIVE AND NEGATIVE WEIGHT COMPUTING

## 3.1 XML Files Preprocessing

For XML files in a corpus, the content of the Denoter tag is marked first (the tag is used to describe the event trigger). Then, the XML file is parsed into uncommented raw news text. After that, the NLP (Natural Language Processing) tool is used to segment sentences and words of the original news text. By using the trigger mark made previously, the index of the trigger in the sentence is recorded, so as to provide the basis for the subsequent judgment of the trigger. Finally, part-of-speech tagging and dependency parsing are performed through the NLP tool. Table 1 shows an example of a preprocessed XML file fragment.

In Table 1, the trigger index is used to record the index value of the trigger. In the example, “死亡” (death) is the trigger word, and its index value is 6. The result of the part-of-speech tagging is one-to-one with the result of the word segmentation. For example, the part of the word “死亡” (death) in the example is “v”, which means the verb. The result of the dependency parsing also corresponds to the result of the word segmentation, but the content corresponding to each word includes two parts: the previous number represents the index of the parent node, and the following English abbreviation represents the dependency between the current word and the parent node. Taking the first “3: ADV” in the instance as an example, the “3” in front of the colon means that the parent node of the word “目前” (currently) is “造成” (caused) with an index value of 3, and the “ADV” after the colon indicates that the dependency relationship between “目前” (currently) and “造成” (caused) is adverbial.

## 3.2 Constructing Feature Set

After the XML files have been preprocessed, the feature set can be constructed. The feature set construction includes three parts: feature set construction of the associated words, the determination of the part-of-speech feature and the determination of the dependency feature.

**Definition 1:** event-triggered associated words, abbreviated as ‘associated words’, include the parent node (the parent node of the head word is empty) and the child nodes of the word in dependency parsing, as well as the left or right positions of the word in the sentence, but exclude punctuation.

First of all, an associated word feature set is constructed from all the words associated with a particular word within a sentence, thereby forming a set of words. If $S_{rel}$ represents the feature set of associated words, $W_{par}$ represents the parent node in dependency parsing, $W_{chi}$ represents the child nodes, $W_{lef}$ represents the left position word, $W_{rig}$ represents the right position word, and $W_{wp}$ represents the punctuation, then the

feature set of associated words can be expressed by formula (1).

$$S_{rel} = \left(W_{par}\mathrm{UW}_{chi}\mathrm{UW}_{lef}\mathrm{UW}_{rig}\right) - W_{wp} \quad (1)$$

Then, the determination of the part-of-speech feature means identifying the part-of-speech of a word as the feature of the word, represented by $P$here.

After that, the determination of the dependency feature means finding the dependency relationship as a feature when a word is found as a child node in dependency parsing, which is represented by $R_{as\_chi}$. Each word can have only one parent node in the dependency parsing of a sentence, that is to say, each word can be a child node only once in the dependency parsing of a sentence. The result of determining the dependency parsing relationship is unique.

Finally, $S_{rel}$, $P$ and $R_{as\_chi}$ are combined to form a feature set $S_{fea}$, thereby completing the construction of the feature set, as shown in formula (2).

$$S_{fea} = S_{rel}\mathrm{U}\{P\}\mathrm{U}\left\{R_{as\_chi}\right\} \quad (2)$$

For example, in Table 1, the associated word feature set $S_{rel}$ of the word "死亡" (die) is {"造成" (cause), "人" (person)}, the part-of-speech feature $P$ is "v", and the dependency feature $R_{as\_chi}$ is "VOB", so the feature set $S_{fea}$ is {"造成" (cause), "人" (person), "v", "VOB"}.

## 3.3 Construction of Trigger Table

The preprocessing of XML files and the construction of feature sets are carried out on the entire corpus, while the trigger table is constructed by processing the training set of the corpus. The trigger table is equivalent to a dictionary in which there are key-value pairs. The key here is the potential event trigger, and the value includes two parts: positive feature set and negative feature set. The contents of these two parts of the value comprise key-value pairs. They are also key-value pairs in both feature sets. The key here is the feature of potential trigger, whose value is the number of times the feature appears in the training set. In the positive feature set, it is the statistics when potential triggers are used as triggers. And in the negative feature set, it is the statistics when potential triggers are used as non-triggers. This is the difference between the positive and negative feature sets.

When the trigger table is constructed, it is processed in sentences. Firstly, each word in a sentence is traversed to determine whether or not it is a trigger according to the index of triggers. Following this determination, the feature in the feature set of the word is added to the corresponding position in the trigger table, and the construction process is completed. The construction of the trigger table is preliminarily completed using the same operation for sentences in all documents in the training set. Finally, the trigger table is pruned to delete the key pairs that do not have positive features. The reason for this is that our approach requires both positive and negative features, it cannot complete event trigger recognition when there are only negative features. The key pairs that do not have positive features in the trigger table should be deleted to remove irrelevant and redundant contents and complete the construction of the trigger table. The trigger table construction algorithm is given as algorithm 1.

**Algorithm 1** The trigger table construction algorithm.

**Input:** preprocessed news texts in the training set *PNT-train*
**Output:** the trigger table *TT*

```
Initialize empty trigger table
For word w in PNT-train
    Construct the feature set S_fea of w
    If w not in TTC (trigger table in construction)
        Use w as a keyword in TTC
    End If
    If the index of w == trigger index
        Add S_fea to the positive feature set part of the
        keyword w as the secondary keyword
        If the secondary keyword already exists
            Increase its value by 1
        Else create the secondary keyword first and initialize
        its value to 1
        End If
    Else similar to steps 8 to 12, the difference is that
    the feature is added to the negative feature set part of
    the keyword w
        End If
    End For
    For keyword kw in TTC
        If the positive feature set part of kw is empty
            Delete kw and its value form TTC
    End If
End For
```

The constructed trigger table contains a number of key-value pairs. The key of each key-value pair is a potential trigger, and the value is the number of times the positive and negative features of the potential triggers appear. Thus, the trigger table can be considered as a statistic for the triggers in the training set and their positive and negative features.

## 3.4 Event Trigger Recognition

After the trigger table has been constructed, the event trigger recognition test can be carried out on the test set of the corpus. The candidate triggers and their feature sets obtained from the preprocessed XML files and the constructed feature sets are matched in the trigger table. According to the matching results and weight computing results, the triggers are judged in terms of event trigger recognition. The positive weight is obtained by formula (3), the negative weight is obtained by formula (4), and the final weight is obtained by formula (5).

$$pos_c = \sum_{i=1}^{n_c} \frac{t_p(c_i)}{1 + t_p(c_i)} \quad (3)$$

$$neg_c = \sum_{i=1}^{n_c} \frac{t_n(c_i)}{1 + t_n(c_i)} \quad (4)$$

$$w_c = pos_c - neg_c = \sum_{i=1}^{n_c} \frac{t_p(c_i) - t_n(c_i)}{\left(1 + t_p(c_i)\right)\left(1 + t_n(c_i)\right)} \quad (5)$$

In formula (3) and formula (4), $pos_c$ and $neg_c$ respectively represent the positive and negative weights of candidate trigger

**Algorithm 2** The event trigger recognition algorithm.

**Input:** preprocessed news texts in the test set *PNT-test*, the trigger table *TT*

**Output:** the result of event trigger recognition in test set

```
For word c in PNT-test
    Set c as the candidate word for event trigger recognition
    If c in TT
        Find the value of the key word c and set it to v, and
        set the positive weight pos_c and the negative weight
        neg_c to 0
        Construct the feature set S_fea of c
        For feature i in S_fea
            Set the feature statistic variable t_p(c_i), t_n(c_i) to 0
            If i in the positive feature set part of v
              the value of the secondary keyword i is assigned
              to t_p(c_i)
            End If
            If i in the negative feature set part of v
              the value of the secondary keyword i is
              assigned to t_n(c_i)
            End If
        End For
        Compute pos_c according to formula (3), compute
        neg_c according to formula (4)
        Compute w_c according to formula (5)
        If w_c >= 0
            c is judged as a trigger
        Else c is judged as a non-trigger
        End If
            Else c is judged as a non-trigger
        End For
```

$c$. $n_c$ represents the number of features in the feature set of the candidate trigger $c$. $t_p(c_i)$ and $t_n(c_i)$ respectively represent the statistical values when the *i-th* feature of the candidate trigger $c$ is successfully matched in the positive feature part and negative feature part of the trigger table. In formula (5), $w_c$ is the difference between $pos_c$ and $neg_c$, that is, the final weight of candidate trigger $c$, which is used to judge whether $c$ is a trigger. Algorithm 2 is the trigger recognition algorithm.

After the event trigger recognition, we need to compare the recognition result with the trigger index obtained from the XML files preprocessing stage. The result of this comparison indicates the level of accuracy of the recognition result, and prepares for the evaluation of the effectiveness of the event trigger recognition.

## 3.5 Corpus and NLP Tool

The corpus used for the experiments in this study is CEC 2.0 (Chinese Emergency Corpus) constructed by the Semantic Intelligence Laboratory of Shanghai University. CEC 2.0 uses XML language as the annotation format to annotate five kinds of emergency news texts, including fires (75 texts), earthquakes (62 texts), terrorist attacks (49 texts), traffic accidents (85 texts) and food poisonings (61 texts). A corpus of 332 annotated texts was obtained. Although CEC 2.0 is small in scale, it has a comprehensive annotation of events and event elements. Event tags are used to represent events, Denoter tags are used to describe triggers of events, and Time, Location, Participant and Object tags are used to describe event elements.

In addition, the NLP tool used in this study is LTP (Language Technology Platform, Che et al. 2010). The tool has been independently developed by the Research Center for Social Computing and Information Retrieval of Harbin Institute of Technology. It offers a set of bottom-up efficient and rich Chinese language processing modules for Chinese natural language texts.

## 3.6 Chinese Event Trigger Recognition Experiments Based on Positive and Negative Weight Computing

### 3.6.1 Setting of Experiments

The programming language used in the experiments is Python 3.6, the corpus is CEC 2.0, and the NLP tool is PYLTP (Python version of LTP). For the experiment, the corpus is divided into a training set and a test set in the ratio of 3:1. The result of event trigger recognition is evaluated by precision $P$, recall $R$ and $F_1$ measures.

### 3.6.2 Comparison Experiment 1

In order to determine the effectiveness of self-defined associated word features on event trigger recognition, we set up the comparison experiment 1. In the experiment, we use the word feature to identify the event trigger as a benchmark. Then different features are added to the trigger table to assist trigger recognition. This completes comparison experiment 1.

The results of comparison experiment 1 are shown in Table 2. As indicated in Table 2, the introduction of dependency feature reduces the recall but improves the precision of event trigger recognition. The final $F_1$ measure is 4.54% higher than the benchmark, indicating that the dependency feature improves event trigger recognition. Compared with the dependency feature, the self-defined associated word feature reduces the recall $R$, but improves the precision $P$, making the final $F_1$ measure increase by 4.81% compared with the benchmark, reaching the highest 68.85% in Table 2. It shows that the associated word feature and dependency feature are similar and both have a good effect on event trigger recognition.

### 3.6.3 Comparison Experiment 2

Comparison experiment 1 shows that the $F_1$ measure is 68.85% by using the feature of associated word to assist trigger recognition, but obviously this result is not ideal. In order to strengthen the role of a single feature in event trigger recognition, this paper proposes a positive and negative weight computing method. To verify the effectiveness of this method, different features are added to the benchmark method, and the trigger recognition is carried out by combining the positive and negative weight computing. We compared this method with the method using the same feature but not combining the positive and

**Table 2** Comparison of results when different features are added to the benchmark.

| Methods | Precision $P$/% | Recall $R$/% | $F_1$ measure/% |
|---|---|---|---|
| Word feature (Benchmark) | 50.73 | 86.85 | 64.04 |
| Benchmark + Position word feature | 74.49 | 54.09 | 62.67 |
| Benchmark + Part-of-speech feature | 52.35 | 86.64 | 65.27 |
| Benchmark + Dependency feature | 61.32 | 77.79 | 68.58 |
| Benchmark + Associated word feature | 68.44 | 69.26 | 68.85 |

**Table 3** Comparison of results by adding different features to the benchmark and combining the positive and negative weight computing.

| Methods | Precision $P$/% | Recall $R$/% | $F_1$ measure/% |
|---|---|---|---|
| Word feature (Benchmark) | 50.73 | 86.85 | 64.04 |
| Benchmark + Position word feature | 74.49 | 54.09 | 62.67 |
| Benchmark + Positive and negative position word feature | 68.38 | 80.27 | 73.85 |
| Benchmark + Part-of-speech feature | 52.35 | 86.64 | 65.27 |
| Benchmark + Positive and negative part-of-speech feature | 76.11 | 72.68 | 74.36 |
| Benchmark + Dependency feature | 61.32 | 77.79 | 68.58 |
| Benchmark + Positive and negative dependency feature | 78.09 | 77.52 | 77.80 |
| Benchmark + Associated word feature | 68.44 | 69.26 | 68.85 |
| Benchmark + Positive and negative associated word feature | 72.55 | 79.13 | 75.70 |

**Table 4** Comparison of the proposed method with the existing event trigger recognition methods.

| Methods | Precision $P$/% | Recall $R$/% | $F_1$ measure/% |
|---|---|---|---|
| Trigger expansion (Zhao et al. 2008) | 54.86 | 69.29 | 61.24 |
| Hybrid neural network (Feng et al. 2018) | 74.20 | 63.10 | 68.20 |
| Our method | 82.01 | 76.51 | 79.17 |

negative weight computing, and completed the comparison experiment 2. The results are shown in Table 3.

It can be seen from Table 3 that after combining the positive and negative weight computing method, the result of all the event trigger recognition methods using a single feature has been greatly improved. The reason is that the method of positive and negative weight computing can take into account the positive and negative aspects and give full play to the role of a single feature in event trigger recognition. Among them, the $F_1$ measure of event trigger recognition method using positive and negative dependency features is the highest, reaching 77.80%. Therefore, the positive and negative weight computing method can effectively improve the results of event trigger recognition after adding different features to the benchmark, and the final experimental results also confirm this point.

#### 3.6.4 Comparison Experiment 3

To further improve the effectiveness of event trigger recognition, we combine the three features of the associated word feature, the part-of-speech feature and the dependency feature. The combination of these three features with positive and negative weight computing is used to recognize event triggers. This is also the method introduced in section 3 of this paper. In order to determine the effectiveness of this method, we compare it with other existing event trigger recognition methods. The results of comparison experiment 3 are shown in Table 4.

Because of the selection of different corpora, different preprocessing methods, and even different selections of documents from the same corpus for the test set and training set, event trigger recognition results will be greatly affected. At present, there is not a standardized, open and unified evaluation system in the field of event trigger recognition, so it is impossible to compare the advantages and disadvantages of each method objectively and fairly, and the effectiveness of only one method can be determined to a certain extent. From the comparison experiment 3, we can see that our method is the best of three evaluation indexes, which is some indication that the result of event trigger recognition based on positive and negative weight computing is ideal.

## 4. APPLICATION OF TEXT CLASSIFICATION BASED ON EVENT TRIGGERS

The research on event trigger recognition based on positive and negative weight computing was introduced earlier. On this basis, we carried out an application based on event triggers: text classification. Currently, many text classification methods are available including naive Bayes (McCallum and Nigam, 1998), EM (Expectation-Maximization, Nigam et al. 2000), neural network (Yang et al. 2018, Howard and Ruder, 2018) and so on. In this paper, the event trigger is used as the feature for text classification, and TFC-ICF and improved chi-square test are used to calculate the feature weight. Finally, the classifier is used for text classification.

### 4.1 Feature Weight Calculation

FOR the purpose of event trigger recognition, we take event triggers as the features of text classification. With the help

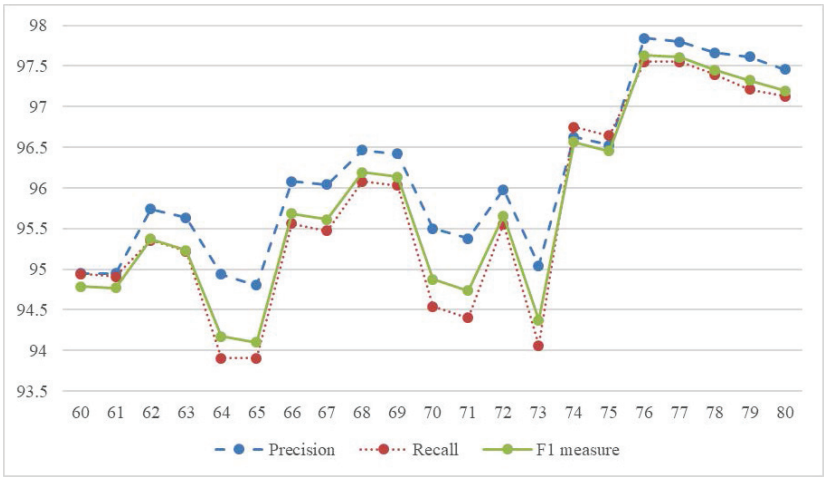


**Figure 2** Classification result when the corpus is divided into training set and test set according to different proportions.

of TF-IDF (Term Frequency-Inverse Document Frequency), a TFC-ICF (Trigger Frequency in Category-Inverse Category Frequency) method is proposed to calculate the feature weights for features and text categories. At the same time, the chi-square value of the feature and the text category obtained by the chi-square test is used as the weight factor to multiply with TFC-ICF to obtain the final feature weight. The calculation method is shown in formula (6). Where *weight(t,c)* represents the feature weight of feature $t$ and text category $c$. $m(c)$ indicates the number of texts in the text category $c$. $f_{dt}$ represents the number of occurrences of feature $t$ in text $d$. $N$ represents the total number of text categories. $n_t$ indicates the number of text categories where feature $t$ has occurred. $\chi^2(t,c)$ is the chi-square value of feature $t$ and text category $c$.

$$\text{weight}(t,c) = \frac{\sum_{i=1}^{m(c)} f_{dt}}{m(c)} \times \frac{N}{n_t} \times \frac{\chi^2(t,c)}{1+\left|\chi^2(t,c)\right|} \tag{6}$$

## 4.2 Classification Method

After calculating the feature weights of all trigger features in a text and each text category, they are accumulated by text category. The accumulation is used as the membership degree of the text and each text category. The text category with the maximum membership degree is regarded as the result of the text classification.

The membership degree *md(d,c)* of text $d$ and text category $c$ can be calculated by formula (7). Where $n$ represents the number of trigger features in text $d$. $weight(t_i,c)$ represents the feature weight of the *i-th* feature $t$ in text $d$ and the text category $c$, which can be obtained from the previous feature weight calculation stage. For the trigger feature $t_i$ that does not appear in the training set, set the $weight(t_i,c)$to 0.

$$md(d,c) = \sum_{i=1}^{n} weight\,(t_i,c) \tag{7}$$

The classification result $r(d)$of text $d$ can be calculated by formula (8). When *md(d,c)* is the maximum, the corresponding text category $c$ is the classification result of text $d$.

$$r(d) = argmax_c\{md(d,c)\} \tag{8}$$

## 4.3 Text classification Experiment Based on Event Triggers

The text classification experiment based on event triggers was performed on CEC 2.0. In order to avoid the cascade error generated by the event trigger recognition method, we directly use the event triggers marked in CEC 2.0 for the experiment. In the experiment, 60% to 80% of the texts in each text category are randomly extracted to form the training set for each text category, and the remaining texts of each category constitute the test set for this category. The average precision $P$, recall $R$ and $F_1$ measures are used to evaluate the overall text classification result.

Figure 2. shows the result of text classification when the corpus is divided into a training set and a test set according to different proportions. The horizontal axis represents the proportion of the training set in the corpus, and the vertical axis represents the percentage of evaluation index. It can be seen that the result of text classification is the best when the training set and test set are divided into 76% and 24% respectively.

In order to evaluate the proposed text classification method more objectively, we compare it with naive Bayes method. Both the methods in this paper and naive Bayes use CEC 2.0 as the corpus, and the proportion of training set to test set is 76%:24%. The results of the comparison are shown in Table 5. According to Table 5, our method has less precision than the naive Bayes method only for the earthquake category, and other evaluation indexes are higher than naive Bayes method, especially the average $F_1$ measure which is 6.94% higher than the naive Bayes result.

## 5. CONCLUSIONS

In order to improve the effectiveness of using word features as a benchmark in event trigger recognition, an event trigger recognition method based on positive and negative weight computing is proposed. On this basis, we study the application

**Table 5** Comparison of the text classification method with naive Bayes method.

| Categories | Precision $P$/% | | Recall $R$/% | | $F_1$ measure/% | |
|---|---|---|---|---|---|---|
| | Naive Bayes | Our method | Naive Bayes | Our method | Naive Bayes | Our method |
| Traffic accidents | 83.33 | 95.45 | 95.24 | 100.00 | 88.89 | 97.67 |
| Earthquakes | 100.00 | 93.75 | 93.33 | 100.00 | 96.55 | 96.77 |
| Terrorist attacks | 92.31 | 100.00 | 100.00 | 100.00 | 96.00 | 100.00 |
| Fires | 87.50 | 100.00 | 77.78 | 94.44 | 82.35 | 97.14 |
| Food poisonings | 92.86 | 100.00 | 86.67 | 93.33 | 89.66 | 96.55 |
| Average | 91.20 | 97.84 | 90.60 | 97.56 | 90.69 | 97.63 |

of text classification based on event triggers. The experimental results show that our event trigger recognition method and its application have achieved excellent results. However, when using multiple features, the event trigger recognition method based on positive and negative weight computing is only a simple combination of methods using a single feature. Therefore, the next step is to combine deep learning with the method using multiple features.

## 6. ACKNOWLEDGMENT

This Research work was supported in part by National Natural Science Foundation of China (Grant No. 62076006), in part by 2019 Anhui Provincial Natural Science Foundation Project (Grant No. 1908085MF189), in part by 2018 Cultivation Project of Top Talent in Anhui Colleges and Universities (Grant No. gxbjZD15).

## REFERENCES

1. Abulaish, M., Sharma, S., and Fazil, M. (2019). A multi-attributed graph-based approach for text data modeling and event detection in Twitter, In: *Proceedings of the 11th International Conference on Communication Systems & Networks*. 703–708.
2. Ahn, D. (2006). The stages of event extraction, In: *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*. 1–8.
3. Alfred, K., Wayne, W., Michael, B., et al. (2018). A knowledge acquisition method for event extraction and coding based on deep patterns, In: *Proceedings of the Knowledge Management and Acquisition for Intelligent Systems-15th Pacific Rim Knowledge Acquisition Workshop*. 16–32.
4. Cao, K., Li, X., Ma, W.C., et al. (2018). Including new patterns to improve event extraction systems, In: *Proceedings of the Thirty-First International Florida Artificial Intelligence Research Society Conference*. 487–492.
5. Che, W.X., Li, Z.H., and Liu, T. (2010). LTP: a Chinese language technology platform, In: *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*. 13–16.
6. Fabio, P., Natraj, R., Timothy, N., et al. (2018). An extensible event extraction system with cross-media event resolution, In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 626–635.
7. Feng, X.C., Qin, B., and Liu, T. (2018). A language-independent neural network for event detection, *SCIENCE CHINA Information Sciences*. 61(9), 092106:1–092106:12.
8. Ferguson, J., Lockard, C., Weld, D.S., et al. (2018). Semi-supervised event extraction with paraphrase clusters, In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 359–364.
9. Ghaeini, R., Fern, X.L.Z., Huang, L., et al. (2016). Event nugget detection with forward-backward recurrent neural networks, In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. 369–373.
10. Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification, In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. 328–339.
11. Huang, L.F., Cassidy, T., Feng, X.C., et al. (2016). Liberal event extraction and event schema induction, In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. 258–268.
12. Huang, L.F., Ji, H., Cho, K., et al. (2018). Zero-shot transfer learning for event extraction, In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. 2160–2170.
13. Laparra, E., Agerri, R., Aldabe, I., et al. (2017). Multi-lingual and cross-lingual timeline extraction, *Knowledge-Based Systems*. 133, 77–89.
14. Li, D.Y., Huang, L.F., Ji, H., et al. (2019). 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*. 1421–1430.
15. Li, M.L., Lin, Y., Hoover, J., et al. (2019). Multilingual entity, relation, event and human value extraction, In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 110–115.
16. Liu, J., Chen, Y.B., Liu, K., et al. (2018). Event detection via gated multilingual attention mechanism, In: *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. 4865–4872.
17. Liu, S.L., Chen, Y.B., Liu, K., et al. (2017). Exploiting argument information to improve event detection via supervised attention mechanisms, In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. 1789–1798.
18. Liu, Y.T., & Li, P.F. (2017). Non-entity event argument extraction on structural representation, In: *Proceedings of the 2017 International Conference on Asian Language Processing*. 306–309.
19. Liu, Z.T., Huang, M.L., Zhou, W., et al. (2009). Research on event-oriented ontology model, *Computer Science*. 36(11), 189–192.
20. Li, W., Cheng, D.Z., He, L., et al. (2019). Joint event extraction based on hierarchical event schemas from FrameNet, *IEEE Access*. 7, 25001–25015.
21. McCallum, A., & Nigam, K. (1998). A comparison of event models for naive Bayes text classification, In: *AAAI-1998 Workshop on Learning for Text Categorization*. 41–48.

22. Nagesh, B.S., N. Satya, K., and Durvasula, V.S. (2017). Event extraction from social media text using conditional random fields, In: *Proceedings of the Working Notes of FIRE 2017-Forum for Information Retrieval Evaluation*. 140–143.
23. Nguyen, T., & Grishman, R. (2015). Event detection and domain adaptation with convolutional neural network, In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. 365–371.
24. Nguyen, T., & Grishman, R. (2018). Graph convolutional networks with argument-aware pooling for event detection, In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*. 5900–5907.
25. Nigam, K., McCallum, A., Thrun, S., et al. (2000). Text Classification from Labeled and Unlabeled Documents using EM, *Machine Learning*. 39(2/3), 103–134.
26. Yang, M., Zhao, W., Ye, Z.Y., et al. (2018). Investigating capsule networks with dynamic routing for text classification, In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 3110–3119.
27. Yang, S., Feng, D.W., Qiao, L.B., et al. (2019). Exploring pre-trained language models for event extraction and generation, In: *Proceedings of the 57th Conference of the Association for Computational Linguistics*. 5284–5294.
28. Zhang, C., Soderland, S., and Weld, D.S. (2015). Exploiting parallel news streams for unsupervised event extraction, *Transactions of the Association for Computational Linguistics*. 3, 117–129.
29. Zhang, T.T., Whitehead, S., Zhang, H.W., et al. (2017). Improving event extraction via multimodal integration, In: *Proceedings of the 2017 ACM on Multimedia Conference*. 270–278.
30. Zhao, Y.Y., Qing, B., Che, W.X., et al. (2008). Research on Chinese event extraction, *Journal of Chinese Information Processing*. 22(1), 3–8.

## 7. DISCLOSURE STATEMENT

No potential conflict of interests was reported by the authors.

## 8. NOTES AND PHOTOS OF CONTRIBUTORS

**Tao Liao** received his Ph.D. degree from the School of Computing Engineering and Science, Shanghai University, Shanghai, in 2014. He is an associate professor at Anhui University of Science and Technology, China. His current research interests include Web mining and semantic search.

**Weicheng Fu** received the Bachelor of Computer Science and Technology, Anhui University of Science and Technology in 2017. Currently, he is a M.S. degree candidate of Computer Science and Technology, Anhui University of Science and Technology, China. His current research interests include natural language processing and data mining.

**Shunxiang Zhang** received his Ph.D. degree from the School of Computing Engineering and Science, Shanghai University, Shanghai, in 2012. He is a professor at Anhui University of Science and Technology, China. His current research interests include Web mining, semantic search, and complex networks.

**Zongtian Liu** is a researcher and doctoral supervisor. His main research fields are program proof and program transformation, advanced language decompilation, software quality management, data mining, rough set, concept lattice, etc.