News Feature Extraction for Events on Social Network Platforms

Peiquan Jin
University of Science and Technology of China
96 Jinzhai Road, Hefei 230027, P.R. China
jpq@ustc.edu.cn

Lin Mu
University of Science and Technology of China
96 Jinzhai Road, Hefei 230027, P.R. China
mulin@mail.ustc.edu.cn

Lizhou Zheng
University of Science and Technology of China
96 Jinzhai Road, Hefei 230027, P.R. China
zhenglz@mail.ustc.edu.cn

Jie Zhao
Anhui University
111 Jiulong Road, Hefei 230601 P.R. China
zj_teacher@126.com

Lihua Yue
University of Science and Technology of China
96 Jinzhai Road, Hefei 230027, P.R. China
llyue@ustc.edu.cn

ABSTRACT
Microblog-based social network platforms like Twitter and Sina Weibo have been important sources for news event extraction. However, existing works on microblog event extraction, which usually use keywords, entities, or selected microblogs to represent events, are not able to extract details of an event. Based on the view of news report, an event should present detailed news features, i.e., when, where, who, whom, and what. Such news features are helpful for conducting deeply data analysis on microblogs, e.g., competitor monitoring and public crisis discovery. However, the challenge is that the news features of an event on microblogs are usually distributed among different posts because of the short-text property of microblogs. This is much different from extracting news events from Web news pages that usually contain most details of an event. In this paper, we propose a new framework to extract events together with their news features from microblogs. We first extract a set of events from microblogs. Each event is represented as a distribution over four kinds of named entities including location, person name, organization, and time. In addition, the type of each event, i.e., location-related, person-related, or organization-related, is determined by a machine-learning method. In order to obtain the news features of an event, we propose an event-clustering approach that puts together all the relevant events into a cluster. For each cluster, we propose different algorithms to extract the news features of the event reported in the cluster. We conduct experiments on two microblog datasets crawled from a commercial microblogging platform to evaluate the performance of the proposed framework. The results suggest the effectiveness of our proposal.

Categories and Subject Descriptors
[**Information Retrieval**]: Retrieval tasks and goals – *Information extraction*

Keywords
Microblog; Event extraction; News features

WWW 2017, April 3-7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3054151

1. INTRODUCTION
Microblog platforms have been one of the major sources for new events detection and spreading. For example, *Sina Weibo* (http://weibo.com) as the most popular microblogging platform in China involves over 280 million users, and over 1,000 tweets are posted every second. Motivated by the massive fresh information generated by microblog users, many works on event detection and analysis over microblogs have been conducted in recent years. These existing works can extract specific types of events [1-3] or events in open domains [4, 5], but they have two problems:

(1) They do not consider event types when extracting events from microblogs. Table 1 shows different types of events. We can see that different types of events have different focuses on the entities embedded in microblogs. For example, location is the central entity of an earthquake, but organization is the focus of a bankruptcy. By distinguishing event types, we can extract the kernel information for an event and further improve the effectiveness of event extraction on microblogs.

Table 1. Different types of events

Event type	Example of events
Location-based	Earthquakes, Fires, Explosions, Riots, Traffic Accidents
Person name-based	Divorce, Arrest, Murder
Organization-based	Bankruptcy, Takeover, Layoffs

(2) They do not provide a fine-grained description for the extracted events. The usual way to represent events is using some selected keywords or named entities [3, 6, 7], or using some selected posts from the original microblogs. However, these approaches are not sufficient for describing events. For example, the words "Na Li" (a Chinese tennis player) and "Champion" are not sufficient to describe the event about Na Li's winning the champion on the Australia Tennis Open at Melbourne on January 2014, because she also won the champion on the French Open at Roland Garros on June, 2011. Therefore, we need to provide more details such as time and location for describing an event [8].

In this paper, we aim at extracting events from microblogs and providing fine-grained representations and details for events on microblogs. We first propose to incorporate event type into event extraction and extract events from microblogs. Each event is represented as a distribution over four kinds of named entities including *location*, *person name*, *organization*, and *time*. In

addition, the type of each event, i.e., location-related, person-related, or organization-related, is determined by a machine-learning method. Then, inspired by the studies in the news-report area that describe an event based on the news features, i.e., *when*, *where*, *who*, *whom*, *what*, and *how*, we consider to detect the news features of events from microblogs.

To extract the news features from microblogs, one basic approach is to select a few representative sentences and then extract the news features from these sentences. However, it is difficult to find all the news features within one microblog, because a single microblog is too short to contain complete information about an event. Moreover, it is difficult to find representative sentences that can cover all aspects of an event, because many people use different texts when describing a same event.

Thus, in this paper we propose an event-clustering approach that puts together all the relevant event units into a cluster. For each cluster, we propose different algorithms to extract the news features of the event reported in the cluster. Briefly, we make the following contributions in this paper:

(1) We propose a new framework for extracting events as well as their news features on microblogs. We first extract a set of typed event units from microblogs, and then cluster the extracted event units to extract the news features of the event within each cluster. Compared with previous works that focus on processing a single microblog, our design is able to gather different aspects of an event and find the details of the event.

(2) We highlight the importance of event type in the process of event extraction. We represent event type as a probability distribution over different named entities and perform a machine-learning method to determine the type of an event.

(3) We present novel algorithms to extract the news features for each event cluster. For extracting *when* and *where*, we introduce a multi-granular similarity-based algorithm. For extracting *who*, *what*, and *whom*, we present a new algorithm based on term clustering and linking.

(4) We conduct experiments on real datasets to evaluate the performance of our proposal. The results show that the proposed framework outperforms existing studies in event clustering and event detail extraction.

The rest of the paper is organized as follows. In Section 2 we summarize the related work. In Section 3, we present the details of event extraction. Section 4 explores the algorithms for extracting the news features. In Section 5, we discuss the experiments and results, and finally we conclude the paper in Section 6.

2. RELATED WORK

In this section, we summarize existing works. There are mainly three research areas that are closely related to our work, i.e., event type determination, event extraction on microblogs, and event detail extraction.

2.1 Event Type Determination

The type of an event, such as natural disaster event, traffic event, or sports event, can be determined according to the nature of the event or the impact of the event. This usually can be done by recognizing different event keywords in texts, e.g., "*earthquake*" and "*forest fire*" for disaster events, and "*goal*!!!" and "*red card*" for sports events. In [4], the researchers proposed an approach based on latent variable models to induce event type which is represented as a distribution over event keywords and specific name entities. However, they did not recognize the type of events when performing event extraction. As finding keywords for all types of events is a tough work, some existing works proposed to focus on one or several specific types of events [1-3].

Differing from the above previous works, in this paper we consider event type from another perspective, i.e., the distribution over named entities such as *location*, *person name*, *organization*, and *time* in an event. Thus, given a microblog dataset extracted by a certain keyword, we can determine the event type of events in the dataset. Then, we can use the event type to improve the extraction of events. Representing an event type using a distribution over named entities has several advantages. First, named entities are crucial for describing an event, because they provide details like time, location, and participants of an event. Second, different events have different distribution over named entities. Thus, we can use the distribution of named entities to distinguish events. Finally, determining an event type using the distribution over named entities is easy to implement because an event involves a limited kinds of named entities according to the properties of a news event.

2.2 Event Extraction on Microblogs

Event detection on microblogs has been a research focus in recent years. Most previous works in this field focused on detecting certain types of events [1-3] or events in open domains [4-5], and many approaches have been proposed such as LDA-based topic modeling [9], text classification and clustering.

Topic modeling is a widely used approach in text mining. The LDA model [9] and its variants are the most representative topic modeling methods. ET-LDA [12] was proposed to model the topics of an event and its associated tweets in a unified framework. Based on ET-LDA, the researchers in [13] performed auto-segmentation of events and classified tweets into two categories, i.e., episodic tweets or steady tweets. In [14], the authors combined an LDA topic model with the recurrent Chinese restaurant process to extract topics and events. Ritter et al. [4] extracted significant events in open domains based on topic models, combining with a named entity tagger and sequence labeling techniques. In [15], the authors first proposed a constrained topic model for tweet representation and then performed fast event monitoring for tweets. Other topic model based approaches were proposed in [16-18]. However, the topic model based approaches need to know the count of topics, which is not reasonable because it is hard to know how many events exist in a microblog dataset. In addition, the properties of microblogs (short length, abbreviations, new words, etc.) make it hard to use the topic models to get high performance for event extraction in microblogs, especially in Chinese microblogs.

Text clustering is another popular approach for extracting events on microblogs. It first extracts features like single words, hashtags [7], *n*-grams, or bursty *n*-grams, and then inputs the extracted features into a similarity-based clustering algorithm extract events. In [11], the authors used the wavelet theory to capture bursty words and performed clustering using the graph partitioning approach. Based on [11] some other works were conducted [6, 10]. The *ET* system [6] utilized a clustering method to extract events. It first extracts event representative keywords in a fixed time interval and then applies a hierarchical clustering technique based on the common co-occurring features of keywords to extract events. *Twevent* [10] detects bursty tweet segments as event segments, which are clustered into events according to their frequency distribution and content similarity. The *STED* system [19] also employs the graph partition-based clustering method to obtain event-related word groups and to generate tweet mini-clusters.

In this paper, we also employ the clustering technique for extracting events. Our approach is based on the agglomerative hierarchical clustering, which has been used in [6] to extract events based on given keywords. Differing from [6], we emphasize the importance of event type (the distribution over named entities) to compute the similarity between microblog posts in each cluster. In

addition, we do not use features like bursty words as they failed to extract events mentioned in few microblog posts [11, 19].

2.3 Event Detail Extraction

Extracting the details of an event is a critical issue in event extraction on microblogs. Previous solutions to this issue can be categorized into three types. The first type uses selected words or phrases. For example, [11] used predefined terms to describe an event, while [10] used phrases. The second type utilizes named entities to describe events [4]. The third type uses a selected set of microblogs to represent events [20-23]. This approach is much popular in online news aggregation and exploration. However, differing from online news, microblogs usually involve a larger volume of data with various styles of representation.

According to human recognition, people want to know the details of an event, such as "who were involved?", "when and where did it happen?", and "what was the essential information about the event?". Previous studies in Web information extraction have focused on extraction time [24-25] and locations [26] from Web pages. These details of an event can be denoted as the news features, which have been widely accepted as the primary metric in traditional news reports [27-28].

Extracting the news features of an event is a sub-task of traditional event extraction task, which is mainly towards news articles. The major task of event extraction has been formulated by MUC [29] and ACE [30]. The goal of ACE's event detection and recognition task is to identify all the event instances, information about the attributes, and the event arguments of each instance of a pre-specified set of event types [30]. In [31], the authors decomposed the ACE task into a series of machine learning sub-tasks. In [32], the authors proposed a scheme to conduct co-reference resolution, cross-lingual and cross-document inference to improve the performance of the ACE's task. In [33], the authors combined event trigger expansion and a binary classifier in event type recognition and utilized the Maximum Entropy method in argument recognition. In recent years, [28] used valency grammar to extract structural semantic information from online news corpus. In [34], the authors extracted the news elements of Chinese news by employing a key event identification algorithm as well as a Semantic Role Labeling (SRL) technique.

Existing works on extracting the news features for events were mainly conducted on news articles. News articles are much different from microblogs with respect to text structure, word formalization, text length, and reporting style. Event extraction on microblogs is a more challenging task because of the special properties of microblogs and the lack of event-related knowledge. Therefore, it is not feasible to regard a microblog as a news article and simply employ previous methods.

3. EVENT EXTRACTION & CLUSTERING

In this section we describe our approach for event extraction. Given a microblog post stream which is related to one specific event query word, we aim to generate microblog post clusters so that posts in a cluster are only correlated with one specific event. Our approach is based on an agglomerative hierarchical clustering method. We highlight the importance of event type in extracting events. In our approach, we first extract event types, which are defined as a distribution over different categories of named entities (i.e., *location*, *people name*, *organization* and *time*) by using machine learning methods. Then, we make use of the event type to calculate similarity between microblog posts and perform an agglomerative hierarchical clustering method to extract events. The architecture of our event clustering methods is shown in Fig. 1.

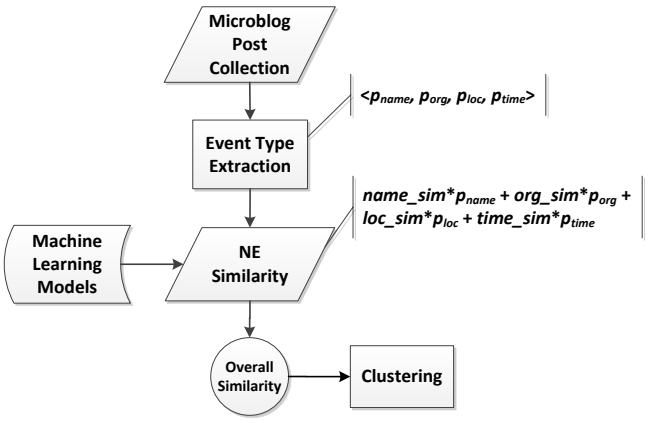

Fig. 1. Architecture of event clustering

3.1 Event Type

Event type is usually described as the nature of an event or the impact of an event, such as natural disaster events, traffic events, and sports events. A common technique to determine the type of an event is to recognize different event keywords (e.g., "*earthquake*", "*forest fire*" for disaster events and "*goal*!!!", "*red card*" for sports events). Event type may play an important role in extracting events since different types of events contain different features such as event keywords. However, there may be too many types of events under this definition and thus extracting event type may have poor performance.

Our definition of event type is based on the distribution over categories of named entities (i.e., *location*, *person name*, *organization* and *time*). We noticed that the named entities distribute diversely over different events. For example, in events like *earthquake* or some other natural disasters, *location* named entities may appear more frequently than other categories of named entities. While in events like *company bankruptcy* or *takeover*, *organization* named entities take a large proportion among all named entities. This diversity in the distribution over different categories of named entities would be a good supplement in extracting events.

Definition 1. ***Event Type.*** *Given a collection of microblog post T which is obtained by one event query word, the event type is defined as a* quadruple *<p_l, p_n, p_o, p_t>, in which p_l, p_n, p_o, p_t represent the importance of location, person name, organization and time entity in the collection respectively. Note that the sum of those four probabilistic values must be equal to one, i.e., $p_l+p_n+p_o+p_t=1$.* □

3.2 Event Type Extraction

In order to make use of event type in extracting events, we first need to extract event type, i.e., the probabilistic quadruple $< p_l, p_n, p_o, p_t >$.

Given a collection of microblog posts T, an intuitive way to extract the quadruple is to calculate the count of appearance of named entities under each category, and divide the total count of appearance of named entities. This simple method will have poor performance because it may incur high variance for microblogs consisting of many noisy data. Thus, we employ a machine learning method to calculate the probabilistic quadruple. We train a Multinomial Logistic Regression classifier that is represented by (1).

$$p_i = p\left(y = i \middle| x^{(i)}, w\right) = \frac{e^{{w_k}^T * x^{(i)}}}{\sum_k e^{{w_k}^T * x^{(i)}}} \quad (1)$$

Here, k=4 and p_i refers to the probability of four categories of named entities p_l, p_n, p_o and p_t. Each collection of microblog posts

are represented as a feature vector *x*, which will be discussed below.

3.2.1 Features

As our method highly depends on named entities, we consider features on named entities in a microblog post collection. Due to the arbitrary writing style of microblog posts, a same named entity may be expressed in different ways. For example, "*Anhui Medical University*") may be expressed as "*AH Med. Univ.*") or "*Anhui Med. Univ.*") or "*Anhui Medical Univ.*"). Thus, we first gather different expressions of the same named entity into one cluster. This process is called *term clustering* (as shown in Fig. 2), which will be detailed in Section 3.3.

The input of the named entity term clustering process is a collection of microblog posts, we perform sentence segmentation and *POS* tagging for each post using *NLPIR*. We then obtain named entities along with there categories (i.e. *location*, *person name*, *organization* or *time*) as the input to the clustering method. The output of the process is a set of clusters in which terms in one cluster represent the same named entity. We also obtain the category of named entity for each cluster. We then extract features in the named entity clusters. Table 2 lists the features for event type extraction.

There are totally 17 dimensions of features, which are divided into five groups. We extract those features based on the statistical indicators under each category of named entities.

First, we define the following symbols. Given a microblog post collection $\boldsymbol{T}$, the set of named entity clusters extracted from $\boldsymbol{T}$ is $\boldsymbol{C^T}$, and the set of named entity cluster under each category is represented as $\boldsymbol{C_l^T}$, $\boldsymbol{C_p^T}$, $\boldsymbol{C_o^T}$ and $\boldsymbol{C_t^T}$. For a cluster set $\boldsymbol{C_i^T}$ (*i*=*l*, *p*, *o* or *t*), $\boldsymbol{C_i^T}[j]$ is the *j*-th named entity cluster in the cluster set.

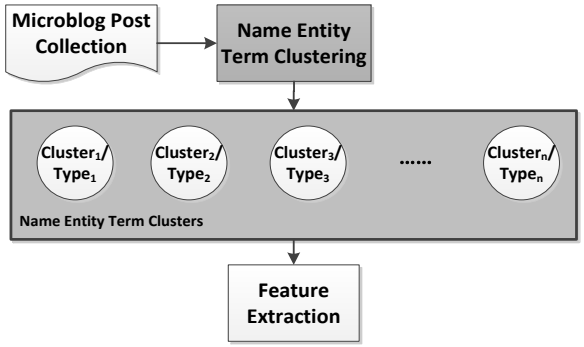

Fig. 2. Input and output of named entity term clustering

Table 2. Features for event type extraction

Group	Dimensions	Description
Group 1	4	Proportion of clusters under each named entities category
Group 2	4	Proportion of named entities under each named-entity category
Group 3	4	Proportion of distinct named entities under each named-entity category
Group 4	4	Entropy of named entity clusters under each named-entity category
Group 5	1	Entropy of named entity clusters for the whole collection

Group 1: This group of features contains the proportion of the count of clusters under each named-entity category, as describe in (2).

$$prop_c[i] = \frac{|\boldsymbol{C_i^T}|}{|\boldsymbol{C^T}|} \tag{2}$$

Here, |*| represents the count of clusters in a cluster set. Each *i* represents a named-entity category.

Group 2: This group of features contains the named entity frequency of a named-entity category in the microblog post collection, as shown in (3).

$$prop_e[i] = \frac{\sum_j |\boldsymbol{C_i^T}[j]|}{\sum_i \sum_j |\boldsymbol{C_i^T}[j]|} \tag{3}$$

Here, $|\boldsymbol{C_i^T}[j]|$ is the count of named entities in cluster $\boldsymbol{C_i^T}[j]$.

Group 3: This group of features is similar with Group 2. The difference lies in that $|\boldsymbol{C_i^T}[j]|$ in (3) is replaced by the count of distinct named entities.

Group 4: We observed that the named entities in different events are likely different, even though they are within the same named-entity category. The named entities appeared in all events are probably not a named entity that is closely related to the event. For example, for an event query word "*Earthquake*", many microblog posts start with the content "*The China Meteorological Administration published that*". In this case, "*China*" is a location named entity and occurs in almost all earthquake related events, but it is not the key location for those events. For dealing with this problem, we propose features based on named entity entropy for each named-entity category, which is described in (4).

$$entropy[i] = -\sum_j \left(\frac{|\boldsymbol{C_i^T}[j]|}{\sum_j |\boldsymbol{C_i^T}[j]|} * log \frac{|\boldsymbol{C_i^T}[j]|}{\sum_j |\boldsymbol{C_i^T}[j]|}\right) \tag{4}$$

The symbols in (4) have the same meaning as those in (3).

Group 5: There is only one feature in this group. This feature is the entropy of named entity clusters for the whole collection. It is similar to the features in Group 4, except that the calculation is performed on the whole microblog post collection.

3.2.2 Extracting Event Type

Given a set of features, we represent a microblog post collection as a feature vector $\boldsymbol{x}$ and then employ the Multinomial Logistic Regression method to train the model, as shown in (4). The result $p_i = p(y = i|x^{(i)}, w)$ where *i*=*l*, *p*, *o* and *t* for different named entity categories is used as the probabilistic distribution.

Next, we perform a smooth process for the probabilistic p_i obtained from the model, as shown in (5).

$$p_i = \frac{\sqrt{p(y = i|x^{(i)}, w)}}{\sum_i \sqrt{p(y = i|x^{(i)}, w)}} \tag{5}$$

After that, we get a quadruple <p_l, p_n, p_o, p_t>, which is outputted as the event type. We will further use this quadruple to perform event clustering.

3.3 Event Clustering

After we have extracted the event type, which is represented as the probabilistic distribution over different categories of named entities, we use it to calculate the similarity between microblog posts, and further perform clustering.

In this paper, we propose to emphasize the importance of named entities when calculating the similarity. Our calculation of similarity consists of two parts, the normal cosine similarity between terms in the microblog posts and the similarity between named entities.

(1) *Normal Term Similarity*: The normal term similarity is based on a bag-of-word model that represents each microblog post as a vector of terms. We employ the basic cosine similarity between two term vectors as the normal term similarity, which is denoted as $\boldsymbol{Sim}_t$.

(2) *Named Entity Similarity*: Another important part of similarity between microblog posts is the named entity similarity. As different categories of named entities may play different roles in different types of named entities, we propose to adjust the weights for the categories of named entities, e.g., to increase the weight of location similarity for *location*-based events like *earthquake,* or to increase the weight of *organization* similarity for *organization*-based events like *enterprise bankruptcy*.

For two microblog posts, we first calculate the named entity similarity under each named-entity category; then we use the event type to weight the importance of each named-entity category. The named entity similarity under each named-entity category is the sum of the similarity for all pairs of named entities that come from different microblog posts under a named-entity category. Specially, for the *m*-th microblog post $M^T[m]$ in a microblog post collection *T*, $E_{i,m}{}^T[j]$ is the *j*-th named entity under named-entity category *i* (*i*=*l*, *p*, *o* or *t*) in microblog $M^T[m]$, and the named entity similarity between two microblog posts $M^T[m]$ and $M^T[n]$ is calculated by (6).

$$Sim_e^T(m,n) = \sum_i \left[p_i * \sum_j \sum_k Sim\left(E_{i,m}^T[j], E_{i,n}^T[k]\right) \right] \quad (6)$$

Here, $\mathrm{Sim}\left(E_{i,m}^T[j], E_{i,n}^T[k]\right)$ is the similarity between two named entities, i.e., the *j*-th named entity in microblog post $M^T[m]$ and the *k*-th named entity in microblog post $M^T[n]$ under named-entity category *i*. This similarity is calculated based on the *Minimum Edit Distance* (*MED*), as shown in (7).

$$Sim\left(E_{i,m}^T[j], E_{i,n}^T[k]\right) = 1 - \frac{MED\left(E_{i,m}^T[j], E_{i,n}^T[k]\right)}{\max\left(E_{i,m}^T[j].length, E_{i,n}^T[k].length\right)} \quad (7)$$

Here, $E_{i,m}^T[j].length$ is the text length of the named entity $E_{i,m}^T[j]$.

(3) *Overall Similarity*: The overall similarity in our clustering process is the weighted sum of the normal term similarity and the named entity similarity, as shown in (8). Here, α is a trade-off between the two similarities, we will experimentally discuss the setting of α in the experiments.

$$Sim = \alpha * Sim_t + (1-\alpha) * Sim_e \quad (8)$$

We employ an agglomerative hierarchical clustering method for event extraction. This clustering approach does not need to first determine the number of clusters. The algorithm starts with merging the posts with the highest similarity score. For two post clusters, we compute the average similarity between posts in the two clusters. The algorithm stops when the similarity between each two clusters are below a threshold. We do not use flat clustering algorithms like K-means because it is difficult to predict the value of the cluster count *K* in advance.

4. NEWS ELEMENT EXTRACTION

4.1 *When* Extraction

Instead of using the posting time of microblogs, we consider the content time and location of microblogs so as to determine the exact temporal information for events. The process to extract *when* is shown in Fig. 3. We focus on two types of time expressions in microblogs, namely absolute time and relative time [23, 24, 35]. The absolute time refers to explicitly represented time expressions which can be directly found in a calendar. For example, "*March* 1, 2014" is an absolute time expression. The relative time refers to those time expressions that need to be further resolved. In this paper, we use the approaches in the previous work [35] to resolve the absolute and relative time expressions in microblogs.

After detecting the absolute and relative time expressions for each collected microblog, we get a set of different time expressions for each event cluster. The next issue is to determine the right time for the event reported in an event cluster. One basic solution is to calculate the frequency of each detected time expression. However, time expressions have various styles, so this method may get low precision. In this paper, we devise a new multi-granular algorithm to determine the right time expressions for events, i.e., to extract the *when* element. We first define four types of time granularities, namely *day*, *half day*, *hour*, and *minute*. Given a time expression *t*, we formulate it in the form of a quadruple <*day*, *half-day*, *hour*, *minute*>. If a time expression does not indicate explicit part in the quadruple, we simply assign a *NULL* value. Table 3 shows some examples of the multi-granular representation of time expressions.

Given the quadruples of the extracted time expressions for each event cluster, we further calculate the frequencies of time elements in each granularity, as well as their co-occurrence in the cluster using a co-occurrence matrix. The process is shown in Algorithm 1. The main idea of Algorithm 1 is that coarser granularities such as *day* are easier to extract and have higher precision since they occur more frequently in a cluster. For a finer granularity time, if it has a high co-occurrence rate with the extracted coarser time expressions, it is likely to be the right time. Thus, we start extracting from the *day* granularity. We finally obtain a result string that contains the finest granularity and most precise time points.

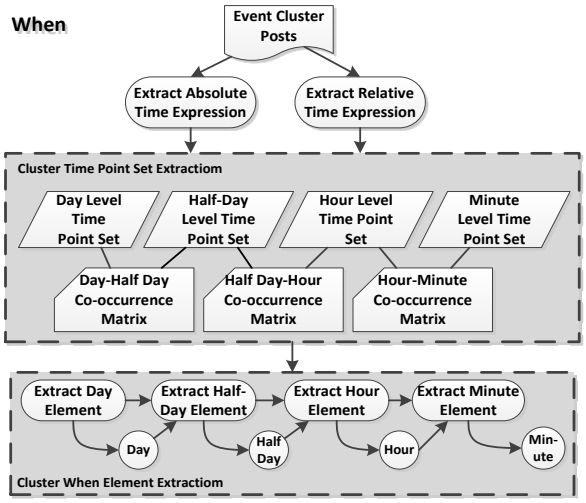

Fig. 3. Process for extracting *when*

Table 3. Examples of the multi-granular representation of time expressions

Time Expressions	Type	Posting Time	Multi-granular Time Representation			
			day	*half-day*	*hour*	*minute*
Evening of Dec 31, 2013/	*Absolute* time	2013-12-31 (*Tue.*)	2013-12-31	*Evening*	*NULL*	*NULL*
10:10 *A.M. of the day before yesterday*	*Relative* time	2013-12-31 (*Tue.*)	2013-12-29	*Morning*	10	10:10
Last Sunday Afternoon	*Relative* time	2013-12-31 (*Tue.*)	2013-12-29	*Afternoon*	*NULL*	*NULL*

ALGORITHM 1 *Event Time Extraction*

Input:

(1) Day-level time point set Dset, Half-Day-level time point set HDset, Hour-level time point set Hset, Minute-level time point set Mset, along with their frequency.

(2) The co-occurrence matrix of each two adjacent levels.

(3) The minimum ratio threshold α and the post count of a cluster N.

Output: *the result time expression resstr.*

```
Get the most frequent day time mdtp in Dset;//day-level
if freq(mdtp)< α*N then return resstr;
else resstr=mdtp;
    Get the most frequent half-day time mhdtp in HDset;//half-day-level
Get the most frequent co-occur day-level time of mhdtp, named mcodtp;
if freq(mhdtp)>= α*N and mdtp==mcodtp then
    resstr +=mhdtp;
else return resstr;
Get the most frequent hour-level time mhtp in Hset; //hour-level
Get the most frequent co-occur half-day-level time of mhtp, named mcohdtp;
if freq(mhtp)>=α*N and mhdtp==mcohdtp then
    resstr +=mhtp;
else return resstr;
Get the most frequent minute-level time mmtp in Mset;//minute-level
Get the most frequent co-occur hour-level time of mmtp, named mcohtp;
if freq(mmtp)>= α*N and mhtp==mcohtp then
    resstr +=mmtp;
else return resstr;
return resstr;
```

4.2 *Where* Extraction

The extraction of *where* is similar to extracting location information from Web pages [36]. In this paper, we define four levels of location granularities, i.e., *province*, *city*, *county*, and *local*. Some location gazetteers are available in the Internet, but in this paper we manually create a Chinese location gazetteer with a hierarchical relation (*province-city-county*/*district*), because the experimental data set is all about Chinese microblogs. We also combine consecutive nouns in a microblog following a location entity to construct the *local* locations. For example, "*Anhui*/ns *Medical*/n *University*/n" will be transformed into "*Anhui*/ns *Medical University*/n", because "*Anhui"* is a province name in China. Thus, "*Medical University*/n" is recognized as a *local*-level location.

4.3 *Who*, *What* and *Whom* Extraction

For the extraction of *who*, *what*, and *whom*, the key problem is the abbreviations and informal structure of microblogs. A named entity may be written in many different textual forms. For example, "*Anhui Medical University*" may be expressed as "*AH Med. Univ*." or "*Anhui Med. Univ*." or "*Anhui Medical Univ*.". To deal with this issue, we present a term clustering method to put different expressions about the same named entity into one term cluster. Besides, we design a novel method to link different term clusters and form <*who*, *what*, *whom*> tuples. Our approach is depicted in Fig. 4.

We first extract initial <*subject*, *predicate*, *object*> and <*subject*, *predicate*> tuples by employing some tailor-made rules like "*NP*1+*VP*+*NP*2" and "*NP*1+*VP*". We combine consecutive nouns for *NP*1 and *NP*2, and combine consecutive verbs for *VP* to obtain longest noun/verb phrases.

Through the initial tuples extraction, we obtain the <*subject*, *predicate*, *object*> tuples. In this step, we aim to refine the tuples. Based on an observation on microblogs, we can find that most of the *who* elements appear in the subject position, and *what* is very possible in the predicate position, while *whom* is more likely appear in the object position. Thus, our basic idea is to first form term clusters for *subject*, *predicate*, and *object*, which results in term cluster sets of *subjects*, *predicates* and *objects* respectively. Next, we link each *subject* cluster with a *predicate* cluster and link each *predicate* cluster with an *object* cluster to form the candidate <*who*, *what*, *whom*> tuples. We conduct clustering for each component of term set by considering textual and co-occurrence information. For textual information, we employ minimum edit distance (*MED*) and longest common subsequence (*LCS*). The textual similarity between two term sets is the average similarity of terms, which is computed by (9), in two term sets. For co-occurrence information, we first get the co-occurrence vector of each term from the co-occur matrix and then compute the cosine similarity between vectors, as shown in (10). The combination of similarity is shown in (11) and parameter β is tuned in our experiment.

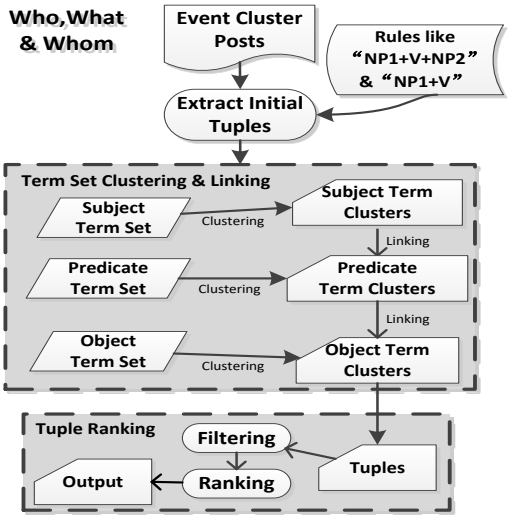

Fig. 4. Extracting *who, what*, and *whom*

Finally, we link clusters to obtain candidate <*who*, *what*, *whom*> tuples. We link each *subject* cluster with a *predicate* cluster, and link each *predicate* cluster with an *object* cluster. The linking process is based on the term co-occurrence frequency. Formula (12) describes the linking process. Given a cluster $c1$ and a set of target clusters, we find a cluster $c2$ in which terms have the largest co-occurrence frequency with the terms in $c1$.

After the clustering and linking step, we got several candidate <*who*, *what*, *whom*> and <*who*, *what*> tuples. In the last step, we rank such tuples to get the final output. We start with extracting the key expression for a term cluster. The key expression is either the most frequent *LCS* between terms in a cluster or a term with quite high frequency. We then rank the tuples based on the average frequency of the key expressions in the tuples. We finally keep K (K=5) tuples as the determined <*who*, *what*, *whom*> or <*who*, *what*> tuples.

$$TEXTUAL_{SIM_{t_1,t_2}} = avg(LCS(t_1,t_2), 1-\frac{MED(t_1,t_2)}{max(t_1.length, t_{2.}.length)}) \quad (9)$$

$$CO\_SIM_{t_1,t_2} = cos(combine(vec_{t_1,M_1}, vec_{t_1,M_2}), combine(vec_{t_2,M_1}, vec_{t_2,M_2})) \quad (10)$$

$$SIM_{t_1,t_2} = \beta * TEXTUAL_{SIM_{t_1,t_2}} + (1-\beta) * CO\_SIM_{t_1,t_2} \quad (11)$$

$$Link(c_1,c_2) = \{(c_1,c_2) | \max_{c2}(\sum_{t_1 \in c_1}\sum_{t_2 \in c_2} freq(t_1,t_2))\} \quad (12)$$

5. PERFORMANCE EVALUATION

5.1 Dataset

To evaluate our methods, we prepare two datasets by crawling posts from *Sina Weibo* (http://weibo.com). Given query words that are related to one or more events, we crawler microblogs that contain those query words. In our work, we consider two heterogeneous data sets: the first dataset consists of posts obtained by crawling posts which contain the given query words. Another dataset is composed of posts describing some specific events. We symbolize the two datasets as *DS*1 and *DS*2, as described below.

***DS*1.** The first dataset *DS*1 contains collections of more than 450K posts crawled by event keywords from Feb 24, 2013 to March 29, 2013. Posts in a collection contain only one specific event keyword. In our experiment, we used 18 event keywords and finally got 18 sets of microblogs.

***DS*2.** Another dataset *DS*2 contains posts about specific events. We obtained those events by searching for posts containing several keywords about the specific event. We totally collected 20 events for evaluation.

5.2 Event Type Extraction

In the training step, we randomly select 10 days from *DS*1, with microblog posts for all query words to construct our training data. Each microblog post collection of a query word in one day is a piece of training data. Thus, we have totally about 180 training data for the 18 queries. We manually label the training data into one of four named-entity category, i.e., *location* based category, *person name*-based category, *organization*-based category and *time*-based category. We test some machine learning techniques to train the model. We use the trained model to test on the remaining 20 days of microblogs for all queries. Since we do not care much about the *recall* metric in this task, we use *precision* to evaluate the method. In our experiment, Multinomial Logistic Regression and Stochastic Decision Tree achieve the best performance, as shown in Table 4.

Table 4. Precision for event type extraction

Technique	Precision
Multinomial Logistic Regression	87.8%
Stochastic Decision Tree	93.2%

5.3 Event Clustering

We consider two types of precision metrics in the evaluation, i.e., the event cluster precision ***PE*** and the overall cluster precision ***PC***. For each cluster we obtained from the clustering step, we manually check all posts in the cluster. We consider a cluster as a true cluster if more than 80% posts in the cluster are related to the same topic. Further on, if the topic in a true cluster is event related, then the cluster is a true event cluster. Formula (14) and (15) describe the two metrics. ***PE*** is the count of true event clusters divided by the count of all clusters we extracted, and ***PC*** is the count of true clusters divided by the count of all clusters. We also evaluate the ***recall*** value of our metric.

$$PE = \frac{\#true\_event\_cluster}{\#all\_cluster} \quad (14)$$

$$PC = \frac{\#true\_cluster}{\#all\_cluster} \quad (15)$$

Figures 5 to 7 show the results of *PE*, *PC* and *recall*, respectively. We evaluate five methods which are shown in Table 5. The difference between those methods lies in the calculation of similarity in clustering. From Figs. 5 to 7, we can see that, for the metric *PE*, our methods (Method 5 and Method 2) achieve a synthesis best performance. However, for the metric *PC* and *recall*, the basic method Method 1 reaches the highest values. After examining the resultant clusters, we discover that for a basic method Method 1, since less restriction is taken on the data when clustering, there are more clusters being aggregated. Thus, the *PC* and *recall* value are higher than methods that based on named entity similarity. However, many clusters in the result cluster set are not event related, which leads Method 1with a worst performance in *PE* metric. Note that for a task of event extraction, event related performance is what we need, i.e. we need high *PE* and *recall* value.

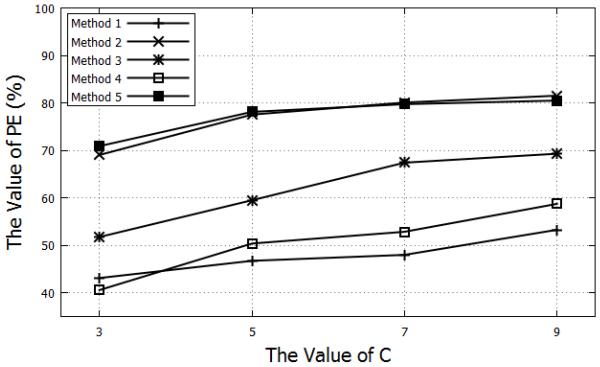

Fig. 5. Result of PE for different methods

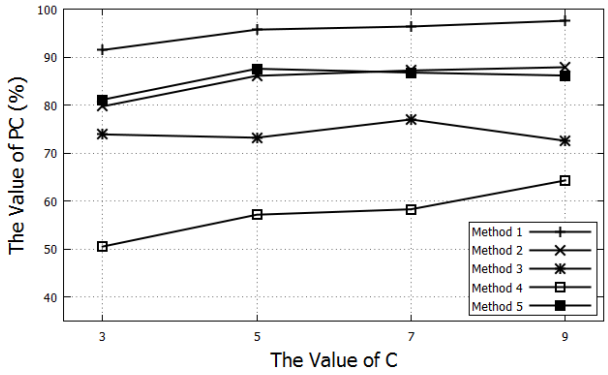

Fig. 6. Result of PC for different methods

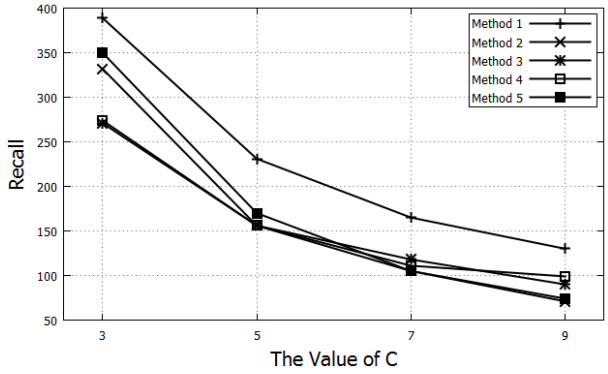

Fig. 7. Result of Recall for different methods

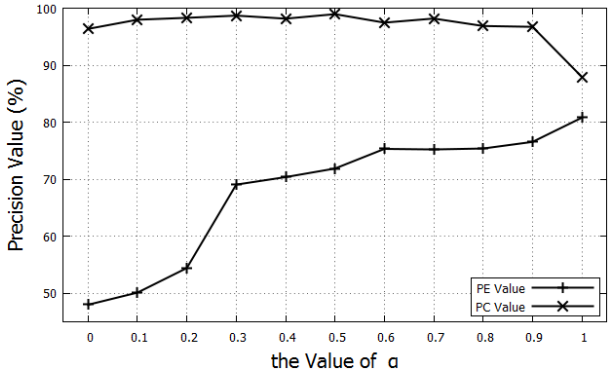

Fig. 8. Result of PE & PC under different values of α

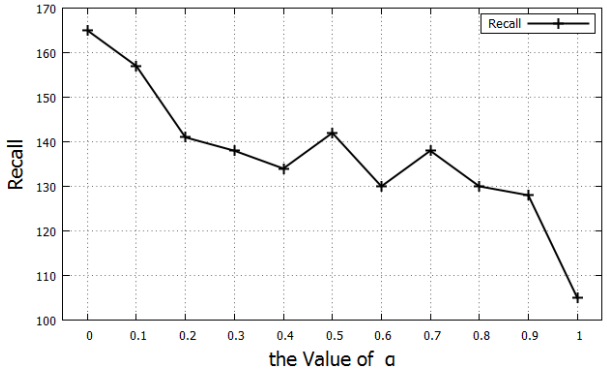

Fig. 9. Result of recall under different values of α

Table 5. Methods for Comparison

Method	Description
Method 1	Consider only normal term cosine similarity.
Method 2	Consider only named entity similarity, all category of named entities share the same weight.
Method 3	Consider only named entity similarity, the weight of each named-entity category is decided by their entity frequency in the post collection.
Method 4	Consider only named entity similarity, the weight of each named-entity category is decided by Formula (1)
Method 5	Consider only named entity similarity, the weight of each named-entity category is decided by Formula (5)

Methods in Table 5 either only consider normal term cosine similarity or only consider named entity similarity. From Figs. 5 to 7, we see that a method of normal term cosine similarity achieves high performance in the metric of *PC* and *recall*, while named entity similarity based methods achieve high performance in the metric of *PE*. So what if we make a trade-off between the two types of similarities? We make this trade-off by adding a weight α between the two similarities, as shown in (8). We compare different values of α in our experiment, Figs. 8 and 9 list the results of precision (*PE* & *PC*) and *recall* under different values of α. Here, we fix the value of *C* to be 7.

5.4 News Feature Extraction

5.4.1 When and Where

The experimental results for *when* and *where* extraction are shown in Table 6 and Table 7. The baseline method is regarding the time and location expression with highest frequency in an event cluster as the result of *when* and *where* extraction. We also compared the results in terms of different granularities.

Given a true event time point, e.g., "9:30 *AM Mar*. 15, 2014", the *day* feature is rightly extracted if we can extract the time expression "*Mar*. 15, 2014" from the event cluster. If the extracted expression is like "8:00 *AM Mar*. 15, 2014", we regard the *day* and *half-day* features are both properly extracted but the *hour* and *minute* features are not extracted.

Table 6. Results of *when* extraction

Dataset		Our Algorithm				Baseline
		Day	Half-Day	Hour	Minute	
***DS*1**	Precision	86.64%	76.92%	69.33%	71.03%	70.90%
	Recall	76.98%	50.36%	37.41%	27.34%	
***DS*2**	Precision	88.89%	73.68%	56.25%	70%	60%
	Recall	80%	70%	45%%	35%	

Table 7. Results of *where* extraction

Dataset		Our Algorithm				Baseline
		Province	City	Country	Local	
***DS*1**	Presicion	80.54%	83.74%	78.18%	53.6%	69.47%
	Recall	53.60%	37.05%	15.48%	24.10%	
***DS*2**	Precision	85%	91.67%	61.54%	71.43%	65%
	Recall	85%	55%	40%	50%	

We define *recall* as the right cluster time points divided by the whole true event cluster number. From Table 6 and Table 7 we see that the *recall* rate decreases as granularity become finer. The reason for this phenomenon is that many events do not contain time or location information for fine granularities, which leads a relatively low *recall* for finer granularities.

5.4.2 Who, What, and Whom

We use two metrics to measure our methods of extracting the *who*, *what*, and *whom* elements.

(1) *Main Tuple Accuracy*. We define *main tuple* as a tuple <*who*, *what*, *whom*> describing the main part of an event.

(2) *Average Tuple Accuracy.* We evaluate the average accuracy of the tuples <*who*, *what*, *whom*> we extracted from all the event clusters. The accuracy is calculated as the count of true tuples divided by all tuples we extracted.

Here, we define *recall* metric as the average count of true tuples we extracted from each event cluster.

Table 8 shows the experiment results of our methods. We compared our method with a method which do not utilize our term clustering & linking method and only conduct rule based method then use the most frequent tuple to extract *who*, *what* and *whom* elements. We can see that our methods improve the results in both *precision* and *recall*.

Since different qualities of clusters may lead to different results, we compare results under different size of clusters. In Fig. 10, we show the results under different size of clusters. We see that the precision slightly decrease when the size of cluster becomes larger, but when the size is extremely large and up to hundreds, our method performs better. This is because that with the increasing of the number of the posts in a cluster, more noise is introduced into the cluster, which slightly decreases the precision. When the posts number is extremely large, the information distribution becomes stable, which benefits our methods.

We also compared the results for different categories of events. The results are shown in Table 9. Here, we manually define three categories of events, which are *location*-based events, *organization*-based events and *human*-based events. *Location*-based events are those highly relied on a geographical place (e.g. *earthquake*, *fire hazard*, etc.). *Organization*-based events are those happened between organizations (e.g. *Enterprise Bankruptcy* of *taken over*, etc.) *Human*-based events are those highly relied on human beings (e.g. *Celebrity divorced*, etc.). We found that *Human*-based events perform worst among the three categories; the reason is that *NER* tools perform badly in recognition Chinese human named entities, which makes semantic analysis difficult.

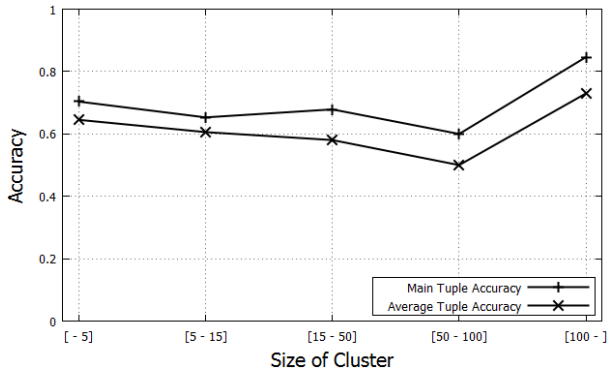

Fig. 9. Accuracy under different size of clusters

Table 8. Results for <*who, what, whom*> tuples

Data Set	Our Algorithm		Baseline	
	Main Tuple Accuracy	Average Tuple Accuracy	Main Tuple Accuracy	Average Tuple Accuracy
***DS*1**	68.71%	61.43%	56.83%	46.11%
***DS*2**	70%	59.26%	45%	45.16%

Table 9. Results of extracting <*who, what, whom*> for different types of events

Event Type	Main Tuple Accuracy	Average Tuple Accuracy
	Precision	Precision
Location-based	75.44%	72.73%
Organization-based	75%	61.67%
Human-based	60.8%	57.03%

6. CONCLUSIONS

In this paper, we aim at providing a mechanism to first extract different types of events from microblogs and then provide a fine-grained semantic analysis on the extracted events. We highlight the importance of the event type in the process of event extraction. We perform machine learning method to determine the type of an event, which defined as the distribution over different named entity categories. We partition microblogs into event clusters based on the types of events. We also introduced some new algorithms to extract the news features for events. Particularly, we present a multi-granular method to extract *when* and *where* information for events. We also proposed a term clustering and linking method to extract the *who*, *what,* and *whom* elements. The experimental results on two real microblog datasets demonstrated the superiority of our methods when compared to the baseline methods.

Our future work will focus on improving the performance of news extraction and try to analyze event evolution [37] using the extracted news features.

7. ACKNOWLEDGEMENTS

This work is partially supported by the National Science Foundation of China (71273010, 613790376, and 61672479). Peiquan Jin is the corresponding author of this paper.

8. REFERENCES

[1] Hogenboom, F., Frasincar, F., Kaymak, U., Jong, F., Caron, E. 2016, A Survey of Event Extraction Methods from Text for Decision Support Systems. *Decision Support Systems*, 85: 12-22

[2] Jang, K., Lee, K., Jang, G., et al. 2016, Food Hazard Event Extraction Based on News and Social Media: A Preliminary Work. *BigComp*, 466-469

[3] Kuzey, E., Vreeken, J., Weikum, G. 2014, A Fresh Look on Knowledge Bases: Distilling Named Events from News. *CIKM*, 1689-1698

[4] Ritter, A., Etzioni, O., and Clark, S. 2012, Open Domain Event Extraction from Twitter, *SIGKDD*, 1104-1112.

[5] Kunneman, F., Bosch, A. 2016, Open-Domain Extraction of Future Events from Twitter. *Natural Language Engineering*, 22(5): 655-686

[6] Parikh, R., and Karlapalem, K. 2013, ET: Events from Tweets, *WWW*, 613-620.

[7] Cui, A., Zhang, M., Liu, Y. et al. 2012, Discover Breaking Events with Popular Hashtags in Twitter, *CIKM*, 1794-1798

[8] Zheng, L., Jin, P., Zhao, J., Yue, L. 2014, A Fine-Grained Approach for Extracting Events on Microblogs. *DEXA*, LNCS 8644, 275-283

[9] Blei, D. M., Ng, A. Y., Jordan, M. I. 2003,: Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3, 993-1022

[10] Li, C., Sun, A., and Datta, A. 2012, Twevent: Segment-Based Event Detection from Tweets, *CIKM*, 155-164.

[11] Weng, J., and Lee, B.-S. 2011, Event Detection in Twitter, *ICWSM*, 401-408.

[12] Hu, Y., John, A., Wang, F. et al. 2012, ET-LDA: Joint Topic Modeling for Aligning Events and Their Twitter Feedback. *AAAI*, 59-65.

[13] Hu, Y., John, A., Seligmann, D. et al. 2012, What Were the Tweets About? Topical Associations between Public Events and Twitter Feeds. *ICWSM*, 154-161.

[14] Diao, Q., Jiang, J. 2013,: A Unified Model for Topics, Events and Users on Twitter, *EMNLP*, 1869-1879

[15] Zhou, X., Chen, L. 2014, Event Detection over Twitter Social Media Streams, *The VLDB Journal*, 23(3), 381-400.

[16] Zhao, J., Li, X., Jin, P. 2012, A Time-Enhanced Topic Clustering Approach for News Web Search, *International Journal of Database Theory and Application*, 5(4): 1-10

[17] Lau, J. H., Collier, N., Baldwin, T. 2012, On-line Trend Analysis with Topic Models: Twitter Trends Detection Topic Model Online, *COLING*, 1519-1534

[18] Zhou, D., Chen, L., He, Y. 2014, A Simple Bayesian Modelling Approach to Event Extraction from Twitter, *ACL*, 700-705

[19] Hua, T., Chen, F., Zhao, L. et al. 2013, STED: Semi-Supervised Targeted-Interest Event Detection in Twitter, *KDD*, 1466-1469

[20] Shen, C., Liu F., Weng, F. et al. 2013, A Participant-Based Approach for Event Summarization using Twitter Streams, *HLT-NAACL*, 1152-1162

[21] Mathioudakis, M., Koudas, N. 2010, TwitterMonitor: Trend Detection over the Twitter Stream, *SIGMOD*, 1155-1158

[22] Cui, T., Zhao, J., Jin. P. 2015, An Efficient Approach to Summarizing Events from Microblogs, *NGCIT*, 19-22

[23] Sharifi, B., Hutton, M. A., Kalita, J. K. 2010, Experiments in Microblog Summarization. *SocialCom/PASSAT*, 49-56

[24] Zhao, X., Jin, P., Yue, L. 2010, Automatic Temporal Expression Normalization with Reference Time Dynamic-Choosing. *COLING*, 1498-1506

[25] Jin, P., Lian, J., Zhao, X., Wan, S. 2008, TISE: A Temporal Search Engine for Web Contents, *IITA*, 220-224

[26] Zhang, Q., Jin, P., Lin, S., Yue, L. 2011, Extracting Focused Locations for Web Pages. *WAIM Workshops*, LNCS 7142, 76-89

[27] Narang, K., Nagar, S., Mehta, S. et al. 2013, Discovery and Analysis of Evolving Topical Social Discussions on Unstructured Microblogs, *ECIR*, 545-556

[28] Wang, W., Zhao, D., Zou, L. et al. 2010, Extracting 5W1H Event Semantic Elements from Chinese Online News. *WAIM*, 644-655

[29] Chinchor, N., and Marsh, E. 1998, MUC-7 Information Extraction Task Definition, *MUC-7*.

[30] ACE. 2017. ACE (Automatic Content Extraction), Chinese Annotation Guidelines for Events. http://www.ldc.upenn.edu/Projects/ACE/docs/Chinese-Events-Guidelines_v5.5.1.pdf

[31] Ahn, D. 2006. The Stages of Event Extraction. *Proc. of COLING/ACL 2006 Workshop on Annotating and Reasoning about Time and Events*, 1-8.

[32] Ji, H., and Grishman, R. 2008, Refining Event Extraction through Cross-Document Inference, *ACL*, 254-262

[33] Hongye, T., Zhao, T., and Zheng, J. 2008, Identification of Chinese event and their argument roles, *Proc. of CIT Workshops*, 14-19

[34] Wang, W. 2012, Chinese News Event 5W1H Semantic Elements Extraction for Event Ontology Population, *WWW*, 197-202.

[35] Lin, S., Jin, P., Zhao, X., Yue, L. 2014, Exploiting temporal information in Web search. *Expert Systems with Applications*. 41(2): 331-341

[36] Zhao, J., Jin, P. Zhang, Q., Wen, R. 2014, Exploiting Location Information for Web Search. *Computers in Human Behavior*, 30: 378-388

[37] Huang, J., Peng, M., Wang, H., Cao, J., Gao, W., Zhang, X. 2017, A Probabilistic Method for Emerging Topic Tracking in Microblog stream. *World Wide Web*, 20(2): 325-350

[38] Alonso, O., 2017, Event Evolution and Archiving, *CIDR*.