Feature-Rich Segment-Based News Event Detection on Twitter

Yanxia Qin[1] **Yue Zhang**[2] **Min Zhang**[3,1] **Dequan Zheng**[1]
[1] School of Computer Science and Technology,
Harbin Institute of Technology, Harbin 150001, China
[2] Singapore University of Technology and Design, Singapore 138682
[3] School of Computer Science and Technology,
SooChow University, Suzhou 215006, China
{yxqin,dqzheng}@mtlab.hit.edu.cn
yue_zhang@sutd.edu.sg zhangminmt@hotmail.com

Abstract

Event detection on Twitter is an important and challenging research topic. On the one hand, Twitter provides first-hand information and fast broadcasting. On the other, challenges include short and noisy content, big volume data and fast-changing topics. Dominant approaches for Twitter event detection model events by clustering tweets, words or segments, while segments have been proven to be advantageous over both words and tweets in news event detection. We study segment-based news event detection, for which existing heuristic-based methods suffer from low recall. We propose feature-based event filtering to address this issue. Our filter incorporate a rich family of features that are empirically proven to be valuable. Experimental results show that our event detection system outperforms the state-of-the-art baseline with doubled recall and increased precision.

1 Introduction

We study news event detection from Twitter messages (tweets). Generally, tweets can be classified into three groups: 1) *news events*, or breaking news such as "Manchester united Vs Athletic in Jan. 1st"; 2) *hot topics* that spread among a large amount of Twitter users, such as horoscope topics (e.g. "You have recently experienced a phase of expansionism and it's... More for Sagittarius"); and 3) *heterogeneous collections* or, meaningless non-event tweets, such as "Need buddy wanna chat". Some previous work (Cataldi et al., 2010; Kasiviswanathan et al., 2011; Diao et al., 2012) regards both news events and hot topics as subjects of detection, while other work (Jackoway et al., 2011; Sakaki et al., 2010; Becker et al., 2012) only detects news events. We are interested in the latter, for which most previous work detects only specific types of events. For example, Sakaki et al. (2010) detect earthquake events from Twitter. In this paper, we study event detection in general.

Compared with event detection in news texts, Twitter provides more opportunities and challenges. Yom-Tov and Diaz (2011) report that Twitter can broadcast news faster than traditional media, which provides an opportunity for event detection in Twitter. On the other hand, there are challenges in event detection from Twitter data: 1) tweets are too short and sometimes cannot carry enough information; 2) tweets contain many noisy words, which can be harmful for event detection and 3) the volume of Twitter data is very large, which makes event detection a big data problem.

The dominant approach for Twitter event detection is clustering. Similar tweets (Becker et al., 2011; Li et al., 2012c) or words (Platakis et al., 2009; Lee et al., 2012) are group into a cluster, before clusters are classified into either news events or non-events. A recent paper (Li et al., 2012a) showed that segments (i.e. ngrams; see Section 2) can be advantageous over both tweets and words for clustering. As segments have much smaller quantity than tweets and are more semantically meaningful than words, they are better units to be clustered. We take Li's system (Twevent) as our baseline system.

Twevent apply a heuristic-based method (*newsworthiness*) to filter out hot topics and heterogenous clusters from news events. *Newsworthiness* is calculated by similarity between edges in a cluster, and whether segments of the cluster frequently appear in Wikipedia. Both similarity of edges and Wikipedia are useful in filtering out heterogeneous collections from news events, while Wikipedia can also separate some news and topics. However, there are several problems with this approach: 1) *newsworthiness* cannot distinguish

International Joint Conference on Natural Language Processing, pages 302–310,
Nagoya, Japan, 14-18 October 2013.

news from some topics, includes horoscope topics ("sagittarius; approach; big trouble") and topics such as "hitler; fox; megan fox; rip; megan; selena gomez", which contain segments that can also frequently occur in Wikipedia; 2) as a single measure, *newsworthiness* is subject to a tradeoff between precision and recall, while a high precision can be obtained only with an extremely low recall (about 10%).

On the other hand, tweets contain useful information that can address the weakness of *newsworthiness*. For example the "Follow spree" topic, which refers to following-back activities by celebrities to their fans, can be recognized by the common hashtag suffix "followspree". Another example is that news tweets are more likely to contain url links. We propose a classifier based method for event filtering and define a set of novel features that capture statistical, social and textual information from event clusters. Some of the features are useful in getting rid of heterogeneous collections while others may be useful for recognizing news events from hot topics.

We call our system Feature-Rich segment-based news Event Detection system on Twitter (FRED). Experimental results show that our system FRED outperforms the state-of-the-art event detection system Twevent by significantly increased precision and doubled recall.

2 Segment-Based Event Detection

In this section, we introduce the segment-based event detection method of Li et al. (2012a), which consists of three steps: tweet segmentation, bursty segment detection and segment clustering. Tweet Segmentation splits tweet into non-overlapping segments, which maybe unigrams or N-grams (2-5 grams). For a certain time window, segments that show a bursty frequency pattern are selected as bursty segments. Segment clustering groups bursty segments about same event into one cluster regarding them as one event.

2.1 Tweet Segmentation

Tweet segmentation can be regarded as a optimization problem to partition tweet with the use of Microsoft Web N-Gram service[1] and Wikipedia[2]. The objective function is defined as:

$$\arg\max_{s_1,...,s_m} C(d) = \sum_{i=1}^{m} C(s_i) \tag{1}$$

where d is a tweet from Tweet stream, $\{s_1, ..., s_m\}$ are the segments in tweet d and C is the function which measures the stickiness of a tweet or segment. In particular:

$$C(s) = L(s) \cdot e^{Q(s)} \cdot S(scp(s)) \tag{2}$$

where the length $L(s)$ is defined in Eq 3, and a longer s makes it typically less sticky. $Q(s)$ is the probability that s appears as an anchor text in Wikipedia articles; frequently-appearing anchor texts are more semantically meaningful. $S(\cdot)$ is the sigmoid funcion. $scp(s)$ is a cohesiveness measurement of segment s defined with symmetric conditional probability, as shown in Eq 4. Better combination of words when forming segments leads to higher cohesiveness value.

$$L(s) = \begin{cases} \frac{|s|-1}{|s|}, & for\ |s| > 1 \\ 1, & for\ |s| = 1 \end{cases} \tag{3}$$

$$scp(s) = \log \frac{Pr(s)^2}{\frac{1}{n-1}\sum_{i=1}^{n-1} Pr(w_1^i) Pr(w_{i+1}^n)} \tag{4}$$

In the above equations, a segment s can be written as $\{w_1 \dots w_n\}(n > 1)$, where $Pr(\cdot)$ is the prior probability derived from the Microsoft Web N-gram service.

2.2 Bursty Segment Detection

From the large number of segments resulting from the last step, a small portion of bursty ones are selected for event clustering since segments with a burst frequency are more representative for a breaking news in the data stream. For convenience, we take a time window t as the time unit for bursty segment detection and segment clustering. N_t refers to the number of tweets within the time window t, and $f_{s,t}$ represents the number of tweets that contain segment s within t. If $f_{s,t} > E[s|t]$, then a segment s is a **bursty segment**. $E[s|t]$ is the expected number of tweets that contain s within t. As N_t is sufficiently large, the Gaussian distribution is used to model the probability of $f_{s,t}$.

$$P(f_{s,t}) \sim N(N_t p_s, N_t p_s(1 - p_s)) \tag{5}$$

[1] http://web-ngram.research.microsoft.com/info/
[2] http://www.wikipedia.org/

where p_s is expected probability of tweets contain s, calculated as:

$$p_s = \frac{1}{L}\sum_{t=1}^{L}\frac{f_{s,t}}{N_t} \quad (6)$$

L is number of time windows containing s.

$E[s|t] = N_t p_s$. Even after filtering out non-bursty segments, a large amount of bursty segments remain. **Bursty weight** $w_b(s,t)$ is assigned to each bursty segments and the top K bursty segments are chosen for further processing. K is set to $\sqrt{N_t}$ in Twevent.

$$w_b(s,t) = P_b(s,t)log(u_{s,t}) \quad (7)$$

$P_b(s,t)$ is the bursty probability and $u_{s,t}$ means the **user frequency** of s helping to filter out some noisy segments, as the more users talk about the segment s, the more popular and meaningful it is. $u_{s,t}$ is calculated as the number of users who post tweets containing s within t.

$$P_b(s,t) = S(10 \times \frac{f_{s,t} - (E[s|t] + \sigma[s|t])}{\sigma[s|t]}) \quad (8)$$

$\sigma[s|t] = \sqrt{N_t p_s(1-p_s)}$ is the standard deviation of Gaussian distribution in Eq 5.

2.3 Segment Clustering

k-Nearest Neighbor graph (kNNgraph) clustering method, is applied to group bursty segments into clusters. The kNNgraph clustering method takes a complete graph of bursty segments with edges representing similarity between segments as input and output event clusters. It groups two segments into same cluster only when they are in each other's k-nearest neighbors. k is a key parameter to control the size of clusters. We choose value for k in Section 4. The output of kNNgraph clustering is an event cluster set corresponding to the time window t, denoted as $Gset(t)$. All $Gset(t)$ sets are gathered to a whole event cluster set $Gset$. $Gset$ is manually labeled for further use, introduced in Section 4.2.

Temporal features and text similarity are incorporated when calculating similarity between two segments s_1, s_2.

$$sim_t(s_1,s_2) = \sum_{m=1}^{M} w_t(s_1,m)w_t(s_2,m)sim(T_1,T_2) \quad (9)$$

$< t_1 \dots t_M >$ are M sub time windows of the time window t. Frequency of segment s in the sub time window t_m is denoted as $f_t(s,m)$. $w_t(s,m)$ is the frequency weight of s in t_m, which serves as a temporal feature and is shown in Eq 10. T_i denotes a set of tweets containing s_i within t_m. $sim(T_1,T_2)$ measures text similarity between the two sets of tweets T_1, T_2. Tweets in T_i are concatenated as a pseudo document, and cosine similarity is applied for calculating distance. Pseudo documents are represented by the Vector Space Model, weighted by TF-IDF. TF value is the number of tweets containing word w within the sub time window t_m and DF value is the number of tweets containing w in the whole twitter corpus.

$$w_t(s,m) = \frac{f_t(s,m)}{\sum_{m'=1}^{M} f_t(s,m')} \quad (10)$$

3 Feature-Rich Event Filter

The clusters in the kNNgraph clustering result $Gset$ contain news events, hot topics and heterogeneous clusters, corresponding to the three types of tweets mentioned in the Introduction. Our goal, which is to recognize news events from other two types of event clusters, is a challenging task because hot topics and news events can both have bursty frequency and share similar characteristics.

3.1 Event Filter in Twevent

Twevent utilizes a heuristic-based method for event filtering using information from Wikipedia. A heuristic equation, *newsworthiness*, is used to determine whether an event cluster is a news event or not, whereas all clusters with a high *newsworthiness* score is news events. The *newsworthiness* $\mu(e)$ of an event cluster e containing segment set S_e and edge set G_e is calculated as follows.

$$\mu(e) = \frac{\sum_{s\in S_e}\mu(s)}{|S_e|} \cdot \frac{\sum_{g\in G_e} sim(g)}{|S_e|} \quad (11)$$

where $\mu(s)$ of segment s is calculated as:

$$\mu(s) = \max_{l\in s} e^{Q(l)} - 1 \quad (12)$$

l is sub-phrase of s and $Q(l)$ is the probability that l appears as anchor text in Wikipedia articles.

An event cluster e is taken as a news event only if it satisfies the condition that $\mu_{max}/\mu(e) < \tau$, where τ is a threshold for *newsworthiness*, μ_{max}

is the maximum of $\mu(e)$ in time window t. Lower τ value leads to high precision and low recall, which is the limitation of *newsworthiness*. Rich information from tweets and clusters themselves can be useful in alleviating this problem.

3.2 Event Filter in FRED

In order to incorporate rich features, we take event filtering as a binary classification problem, where class 'T' means true news event class includes news events and class 'F' represents false news event class containing hot topics and heterogenous clusters. In our filter, event clusters in $Gset$ are represented with a set of cluster-level features, and classified into T or F by a SVM[3] classifier. All clusters in class T, represented as $Eset$, form the final news event result. Features used to represent event clusters are shown in Section 3.3.

3.3 Features

We collect three types of features for the filter, representing statistical, social and textual information related of event clusters, respectively. Some of the features are designed to filter out heterogeneous clusters, while others to distinguish news events from hot topics. Given an event cluster e and the corresponding time window t (from which e is extracted), we have the following information: 1) $Gset(t)$, a sub set of $Gset$ corresponding to t. 2) S_e, the set of segments in e and G_e, the set of edges in e. 3) $T(e)$, which consists of tweets that are related to e containing at least one segment of S_e and being posted in t. 4) $relU(e)$, which represents users who posted the tweets in $relT(e)$, and U_t, which denotes the number of users who published tweets within t.

Statistical Features

For statistical features, we collect direct statistical information from event clusters, such as how many segments and edges it contains, the density of the event graph and so on.

- *seg*, which refers to the segment number of e, calculated as $|S_e|/\max_{e' \in Gset(t)}(S_{e'})$. News events and hot topics contain more segments than heterogeneous clusters generally.

- *edge*, which refers to number of the edges of e, defined as $G_e/\max_{e' \in Gset(t)}(G_{e'})$. Similar with segment number, heterogeneous clusters usually have less edges than news and topics.

- *wiki*, the average of *newsworthiness* for all segments in S_e. A higher *wiki* value indicates that the event cluster contains more meaningful and important segments. *wiki* is able to distinguish news events from hot topics and heterogeneous clusters, as shown Li et al. (2012a).

- *dup* is designed to filter out some specific heterogeneous clusters that contains words sharing the same lemma. For example the event cluster e7 in Table 3, which words sharing lemma "feel". *dup* can be obtained by stemming all unigrams appeared in S_e and calculating the number of duplicated stemmed unigrams out of all stemmed unigrams.

- *sim*, which refers to average similarity of all edges in G_e. A bigger *sim* means that the event cluster is more dense, or sticky.

- *df*, which refers to the number of tweets related to e out of all tweets published in t, namely $|relT(e)|/N\_t$. *df* could help to eliminate heterogeneous clusters, which are published by less users in less tweets.

- *udf*, which refers to $|relU(e)|/U\_t$. The influence of *udf* is similar with *df*.

Social Features

Tweets in $relT(e)$ contain rich Twitter-specific social information, which may reveal the difference between news events and hot topics. For example, the more mentions (@username) exist in $relT(e)$, the more likely e is a topic.

- *rt*, which represent how many tweets in $relT(e)$ are retweeted. *Retweet* is a forwarding action on a tweet published by other users indicating an interest to the tweet. A retweeted tweet is denoted by a prefix of "RT @username". *Retweet* functions as a means of sharing and spreading without commenting to show user's opinion. A news event may have a larger fraction of retweeted tweets than others as users want to spread the news.

- *men*, which refers to the normalized number of tweets containing *mention* (e.g., @username) in $relT(e)$ specifying one target

[3] We use LibSVM http://www.csie.ntu.edu.tw/ cjlin/libsvm/ for the experiment.

receiver of tweets (e.g., "@justinbieber"). *Mention* actions occur more frequently in hot topics than in news events, as users prefer showing their opinion about this topic rather than just spreading it.

- *rep*, which refers to the normalized number of reply tweets in $relT(e)$. *Reply* means commenting, and a reply tweet is started with a mention. Similar with *mention*, *reply* has strong indication of conversation, and are more related to topics than news.

- *url*, which refers to the normalized number of tweets containing url link in $relT(e)$. *Url* shows extra information for tweet. News events contains more information than a topic, which may not be fully expressed in a short tweet, and hense url links are likely used to refer to the original article.

- *tag*, which refers to the normalized number of tweets containing *hashtag* in $relT(e)$. A *hashtag* (#gamecocks) is a short description of what's happening. Generally a popular hashtag indicates a hot topic or an event (e.g., "The game got a little exciting today but we got the win! #gamecocks").

- *pst*, which measures how many tweets contain words in past tense in $relT(e)$, normalized by $|relT(e)|$. News events are more likely to be described formally and with more words in past tense.

Textual Features

Besides above groups of features, text information embedded in hashtag content are another valuable source of information. News events will more likely have common hashtags. For example, many tweets about "National Football League" games have a common hashtag "#NFL". Twitter topic can have common prefixes or suffixes of hashtag. For example the "Follow spree" topic, which is mentioned earlier, may have a common hashtag suffix "followspree".

- *fTag*, which represents how many hashtags appear in $relT(e)$ are frequent hashtags. We extract a frequent hashtag list from whole Twitter data set by takeing the top 2000 most frequently used hashtags.

- *psfx*. To obtain frequent hashtag prefixes/suffixes, we first filter out prefixes/suffixes of all hashtags in the data that satisfy at least one of the following conditions: 1) less than 3 characters, 2) composed by repeating one character, 3) frequency lower than 200. After arranging the prefixes in alphabetical order, we keep only the longest prefixes for the same prefix pattern. Prefixes are ranked by frequency, and the top 2000 are taken as frequent hashtag prefixes. Similarly, we could extract 2000 most frequent hashtag suffixes. *pfx* and *sfx* are used to indicate how many hashtags tweets in $relT(e)$ contain frequent prefixes or suffixes respectively. *psfx* is the combination of *pfx* and *sfx* by multiplying them.

4 Experiments

4.1 Data

The Twitter data we use were crawled from Twitter timeline, which is the real-time tweet stream containing all tweets published by Twitter users from January 1st to January 15th, 2013. After removing stops words, filtering out non-English tweets and null content tweets, the data set contains 31,097,528 tweets published by 16,331,133 users with 382,475 words.

Wikipedia data is used as an extra resource in the tweet segmentation tasks (Section 2.1) and event filtering (Section 3). We use the Wikipedia dump data[4] of February 4th, 2013. It includes 13,167,739 pages and 10,507,127 anchor entities that have 5 words length limit. These anchor entities' anchor probability, i.e. the number of pages that entity e appears as anchor text divided by the number of pages containing entity e, are calculated at the very beginning.

4.2 Settings

We reproduced Twevent as our baseline system. Parameter τ in Twevent and $gamma$ in FRED are tuned for best performance on a development set, which consists all event clusters on Jan. 2nd and 5th. Time window t is set to be a day and M (in Eq. 9) is 12. k in kNNgraph clustering method is set to be 5, as a tradeoff of the number of event clusters and average number of segments in clusters. τ in Twevent is tuned and set to be 2 and $gamma$ in LibSVM of FRED is 5. 10-fold cross validation is

[4] http://burnbit.com/download/235406/enwiki_20130204_pages_articles_xml_bz2

ExpID	FeatureSet	Precision	Recall	F1	Diff
0	All	83.64%	22.89%	35.94%	-
1	All-{seg}	82.73%	22.64%	35.55%	-0.39%
2	All-{edge}	82.35%	22.64%	35.51%	-0.43%
3	All-{df}	83.26%	22.89%	35.9%	-0.04%
4	All-{udf}	83.26%	22.89%	35.9%	-0.04%
5	All-{wiki}	78.57%	17.79%	29.01%	-6.93%
6	All-{dup}	82.88%	22.89%	35.87%	-0.07%
7	All-{sim}	77.78%	17.41%	28.46%	-7.48%
8	All-{rt}	82.51%	22.89%	35.83%	-0.11%
9	All-{men}	83.33%	22.39%	35.29%	-0.65%
10	All-{rep}	81.28%	22.14%	34.8%	-1.14%
11	All-{url}	82.38%	21.52%	34.12%	-1.82%
12	All-{tag}	82.35%	20.9%	33.33%	-2.61%
13	All-{pst}	83.78%	23.13%	36.26%	+0.32%
14	All-{ftg}	81.9%	22.51%	35.32%	-0.62%
15	All-{psfx}	83.56%	22.76%	35.78%	-0.16%

Table 1: Experimental Results Using Different Features.

utilized to get system-generated class labels for all event clusters.

We built a standard gold set for FRED after labeling the event cluster set $Gset$, which is the output of the segment-based event detection (Section 2). The labeling method is shown as follows. Given an event cluster e, the segments in e and the corresponding time window t, we use the segments and t to determine whether e is related to a news. Google and Twitter search are used to assist mannual annotations of events. As a result, 4249 event clusters in $Gset$ were manually labeled into 804 news events and 3445 non-events. Note that some news events in $Gset$ may be sub events of one event. For example "The Golden Globe Awards ceremony 2013" happened in January 13th are detected more than once, as people talked about winners for different awards. We have not merged these sub events in this paper, which will be considered for future work.

With the event cluster set $Gset$, we use the precision, recall and F1-measure to evaluate the performances of FRED and Twevent, where precision is defined the fraction of news events in system-generated 'T' class event clusters ($Eset$ for FRED), and recall measures how many manually labeled news events are detected out of all news events in $Gset$. F1 measure is calculated for an overall evaluation. Note that given our annotations, which is much larger than that of Li (Li et al., 2012a), we can give a better estimation of recall, which Li et al. were not able to report in detail (they used the number of detected news as recall, which did not reveal the real recall notion).

4.3 Experimental Results and Analysis

As we show some statistical results of tweet segmentation (Section 2.1), we obtained 1,604,129 distinct segments with 22.3% unigrams, 72% 2-grams and 5.7% 3-5 grams.

Effectiveness of Features

In Table 1 we show the results of feature ablation test. ExpID is the experiment id. FeatureSet is the features we used for current experiment. All means all statistical, social and textual features. Diff means the difference between F1 in current experiment and experiment 0, and a smaller Diff indicates that the feature is more valuable.

The experimental results show that nearly all features contribute to event filter on either precision or recall. Features can be partitioned into three groups according to their impact on precision and recall: 1) features that are useful only for precision include *df*, *udf*, *dup*, *rt*. 2) features that are useful for both precision and recall includes rest of features such as *wiki*, *sim*, *url* etc. 3) feature *pst* is slightly harmful for precision and recall.

The most valuable features to our system are *wiki*, *sim*, *rep*, *url* and *tag*. *wiki* is extra resource obtained from Wikipedia, and contributes to valuable segments in event clusters. *sim* indicates denser event cluster with stronger connections between segments, while replied tweet number, url number and hashtag number are social features embedded in tweets related to event clusters. Results show there are bigger differences in these features between news events and others clusters when compared to other social features.

The Performance of FRED

System	#Evt	P	R	F1
Twevent$_u$	114	68.42%	9.7%	16.99%
Twevent	107	75.70%	10.07%	17.78%
FRED	146	83.64%	22.89%	35.94%

Table 2: Experimental Results.

The experimental results of FRED and baseline systems are presented in Table 2. Twevent$_u$ is a variant of Twevent, which uses unigrams (words) instead of segments in the event detection. #Evt is the number of news events.

The experimental result of Twevent (precision 75.7%) is lower than that reported by Li et al. (2012a) (precision 86.1%). It is likely to be caused by 1) different Twitter data, Li use Singapore Twitter data containing 4.3 million tweets in one month while ours is global Twitter data of 31.1 million tweets in half a month; 2) horoscope topics are very popular in our data, which cannot be filtered out by Twevent. Because horoscope topics greatly influence the performance of Twevent, we performed a manual filtering to them for a better result. Twevent without the extra process yields 125 event clusters with a low precision of 64.8%. No extra filtering process was necessary for FRED.

The results in Table 2 show that, 1) segments are better than words for news event detection as Twevent outperforms Twevent$_u$, which brings in more heterogeneous collections; 2) our system FRED performs better than Twevent with significantly increased precision and doubled recall, which proves that feature-rich event filter could alleviate the low recall problem in Twevent.

Analysis

We show some example event clusters in Table 3. Lm refers to manually annotated class label. Lt and Lf refer to class labels generated by Twevent and FRED, respectively. The labeling results in Table 3 show that Twevent and FRED made different types of mistakes. As mentioned earlier, Twevent (without manual filtering) always fails to distinguish horoscope topics, while FRED can. From e2-e3 and e4-e5, we can also see that Twevent's labeling result changes for same news events while FRED gives consistent labels. Note that one important difference between FRED and Twevent is that the former uses some supervision. Preliminary experiments show that unsupervised clustering such as k-means clustering cannot effectively bring the benefits of rich-features.

Football and basketball games, which appear almost everyday, take a large fraction of news events. Events such as the 27th Golden Globes Award ceremony hosted on January 13th, show bursty frequency patterns from late January 13th to 14th. Topics such as horoscope topics are popular everyday. At least from our data, the most popular hobbies of the globe seem to be football games.

Among all events, concert news or gossips about celebrities such as "Justin Bieber" and "Taylor Swift" draw much more and much longer attention. For example, e4 in Table 3, which is a news event related to Justin Bieber, continues to appear as news in many days very longer than e1 (a news related to song). New episode of TV programs and TV series such as "Big Brother" and "Pretty Little Liars" are also popular news events.

5 Related Work

Document-pivot clustering methods are frequently used in event detection on social media, in which short messages are regarded as documents (Becker et al., 2011; Li et al., 2012c). Becker et al. (2011) represent text content of a tweet as a TF-IDF weight vector and apply an incremental clustering algorithm to group similar tweets with one cluster regarded as an event. In the following event classification phase, temporal, social, topical and Twitter-centric features are used to represent each cluster and clusters are determined whether they are event-related or topic-related or non-event.

As social media data is on an extremely big scale, document-pivot clustering methods are ineffective as they are time- and memory-consuming. In contrast, in feature-pivot clustering methods, only features (words) that show a burst frequency pattern in a time window are extracted and then clustered into groups to get events. In addition to improving clustering efficiency, detecting bursty features also plays an important role for feature selection as social media messages are very noisy.

In most feature-pivot clustering methods, events are represented as a few representative words showing what happened, which may cause events to be difficult to understand (Li et al., 2012a; Platakis et al., 2009; Lee et al., 2012; Fung et al., 2005). Li et al. (2012a) adopted tweet segmentation in their event detection system Tweven-

ID	Lm	Lt	Lf	Time	Segments	Detail
e1	T	T	T	15th	golden disk awards; 27th; cr; kris; preview	Golden Disk Awards
e2	T	T	F	4th	lead; fans; check; vote; favorite music; peoples choice	peoples choice voting
e3	T	F	F	5th	lead; fans; check; vote; favorite music; peoples choice	related to e4
e4	T	T	T	2nd	paparazzi; chaos; accident; dangerous; fools; princess di	photographer died when chasing justin bieber
e5	T	F	T	3rd	paparazzi; town; sm; went	related to e6
e6	F	T	F	12th	venus; amorous; squares edgy	horoscope topic
e7	F	F	F	7th	feel; feel bad; feel i'm; feel sick	heterogeneous collection

Table 3: Example Events.

t. Tweet segmentation is firstly proposed by Li et al. (2012b) for an named entity recognition system on Twitter. They claim that segments are much more meaningful and easier to read than words. Twevent is the most related work to this paper. We adopt tweet segmentation, and segment tweets into non-overlapping segments that are regarded as bursty feature candidates, and utilize a feature-pivot clustering method to group bursty segments into clusters as events. The difference between this paper and Twevent is that they use a simple measurement (*newsworthiness*) to filter out meaningless twitter topics from events, while we propose a classifier based filter to distinguish news events and twitter topics. The advantage of our system is that it supports the definition of rich features, some of which are helpful to eliminate heterogeneous clusters and others can distinguish news events and hot topics. We will explore their functions in this paper.

In addition to the above group of work, which represents events with a few messages or features showing the topic information, some researchers try to extract structured information for events. Given a set of seed events, Benson et al. (2011) use a factor graph to extract artist and venue information of a concert event. Popescu et al. (2011) extract main entities, actions and audience opinions.

Data from social medias like Twitter are very sparse in presenting thousands of events, while some researchers mainly focus on specific types of events. Sakaki et al. (2010) detected disaster events like earthquakes and typhoons from Twitter. Pohl et al. (2012) tried to detect subevent to assist disaster management with Flickr and YouTube data. Agarwal et al. (2012) analyzed tweets containing specific keywords and report Fire-in Factory and Labor-Strike events. They have fixed query words and search for related messages from social media websites for data. The query words are challenges to define as they are vital to the quality of dataset, which will greatly influence the results. Becker et al. (2012) tried to generate queries for a planned event to relax the limitation. Our work mainly focus on news event detection problem on Twitter.

Rich features have been used in other tasks in NLP, such as POS-tagging (Toutanova et al., 2003), parsing (Zhang and Nivre, 2011) and machine translation (Chiang et al., 2009). Our work is in line with these.

6 Conclusion

We proposed a feature-rich classifier to recognize news events for segment based event detection, defining novel statistical, social and textual features for the filter. Experiments showed the effectiveness of the method, and in particular some features such as the number of urls and hashtags. The feature-rich event filter led to significantly higher precision and doubled recall when compared to the state-of-the-art baseline system. In our experiments we observed that a news event can be detected more than once in one time window, which each appearance representing one aspects of the event. Building these sub-events into a hierarchy will be explored in the future.

Acknowledgments

Yanxia Qin is supported by SRG-SUTD2012038 from Singapore University of Technology and Design and the National Natural Science Foundation of China (No. 61073130) from Harbin Institute of Technology. Yue Zhang is fully supported by SRG-SUTD2012038.

References

Puneet Agarwal, Rajgopal Vaithiyanathan, Saurabh Sharma, and Gautam Shroff. 2012. Catching the long-tail: Extracting local news events from twitter. In *ICWSM*.

H. Becker, M. Naaman, and L. Gravano. 2011. Beyond trending topics: Real-world event identification on twitter. In *Fifth International AAAI Conference on Weblogs and Social Media*.

Hila Becker, Dan Iter, Mor Naaman, and Luis Gravano. 2012. Identifying content for planned events across social media sites. In *Proceedings of WSDM*, pages 533–542, Seattle, Washington, USA.

Edward Benson, Aria Haghighi, and Regina Barzilay. 2011. Event discovery in social media feeds. In *Proceedings of ACL-HLT*, pages 389–398, Portland, Oregon.

Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. 2010. Emerging topic detection on twitter based on temporal and social terms evaluation. In *Proceedings of KDD*, pages 4:1–4:10, Washington, D.C.

David Chiang, Kevin Knight, and Wei Wang. 2009. 11,001 new features for statistical machine translation. In *Proceedings of NAACL*, pages 218–226, Boulder, Colorado.

Qiming Diao, Jing Jiang, Feida Zhu, and Ee-Peng Lim. 2012. Finding bursty topics from microblogs. In *Proceedings of ACL*, pages 536–544, Jeju Island, Korea.

Gabriel Pui Cheong Fung, Jeffrey Xu Yu, Philip S. Yu, and Hongjun Lu. 2005. Parameter free bursty events detection in text streams. In *Proceedings of VLDB*, pages 181–192, Trondheim, Norway.

Alan Jackoway, Hanan Samet, and Jagan Sankaranarayanan. 2011. Identification of live news events using twitter. In *Proceedings of LBSN*, pages 25–32, Chicago, Illinois.

Shiva Prasad Kasiviswanathan, Prem Melville, Arindam Banerjee, and Vikas Sindhwani. 2011. Emerging topic detection using dictionary learning. In *Proceedings of CIKM*, pages 745–754, Glasgow, Scotland, UK.

Sungjun Lee, Sangjin Lee, Kwanho Kim, and Jonghun Park. 2012. Bursty event detection from text streams for disaster management. In *Proceedings of WWW Companion*, pages 679–682, Lyon, France.

Chenliang Li, Aixin Sun, and Anwitaman Datta. 2012a. Twevent: segment-based event detection from tweets. In *Proceedings of CIKM*, pages 155–164, Maui, Hawaii, USA.

Chenliang Li, Jianshu Weng, Qi He, Yuxia Yao, Anwitaman Datta, Aixin Sun, and Bu-Sung Lee. 2012b. Twiner: named entity recognition in targeted twitter stream. In *Proceedings of SIGIR*, pages 721–730, Portland, Oregon, USA.

Rui Li, Kin Hou Lei, Ravi Khadiwala, and Kevin Chen-Chuan Chang. 2012c. Tedas: A twitter-based event detection and analysis system. In *Proceedings of ICDE*, pages 1273–1276, Washington, DC, USA. IEEE Computer Society.

Manolis Platakis, Dimitrios Kotsakos, and Dimitrios Gunopulos. 2009. Searching for events in the blogosphere. In *Proceedings of WWW*, pages 1225–1226, Madrid, Spain.

Daniela Pohl, Abdelhamid Bouchachia, and Hermann Hellwagner. 2012. Automatic sub-event detection in emergency management using social media. In *Proceedings of WWW Companion*, pages 683–686, Lyon, France.

Ana-Maria Popescu, Marco Pennacchiotti, and Deepa Paranjpe. 2011. Extracting events and event descriptions from twitter. In *Proceedings of WWW*, pages 105–106, Hyderabad, India.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of WWW*, pages 851–860, Raleigh, North Carolina, USA.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of NAACL*, pages 173–180, Edmonton, Canada.

Elad Yom-Tov and Fernando Diaz. 2011. Location and timeliness of information sources during news events. In *Proceedings of SIGIR*, pages 1105–1106, Beijing, China.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the ACL-HLT*, pages 188–193, Portland, Oregon.