World Wide Web (2018) 21:609–627
DOI 10.1007/s11280-017-0477-6

Event summarization for sports games using twitter streams

Yue Huang[1] · Chao Shen[2] · Tao Li[1,2]

Received: 5 April 2017 / Revised: 24 June 2017 / Accepted: 27 June 2017 /
Published online: 8 July 2017

Abstract Given a textual data stream related to an event, social event summarization aims to generate an informative textual description that can capture all the important moments, and it plays a critical role in mining and analyzing social media streams. In this paper, we present a general social event summarization framework using Twitter streams. The proposed framework consists of three key components: participant detection, sub-event detection, and summary tweet extraction. To make the system applicable in real data, an online clustering approach is developed for participant detection and an online temporal-content mixture model is proposed to conduct sub-event detection. Experiments show that the proposed framework can achieve similar performance with its batch counterpart.

Keywords Social event summarization · Participant detection · Temporal mixture model · Online update

1 Introduction

1.1 Background

Thousands of events are being discussed on the social media websites everyday [2, 5, 16, 19]. Using the social media, people report the events they are experiencing or publish comments in real-time, which are aggregated into a highly valuable stream of information that informs us the events happening around the world. For instance, Twitter, one of the most

✉ Tao Li
taoli@cis.fiu.edu

[1] Jiangsu BDSIP Key Lab, School of Computer Science, Nanjing University of Posts and Telecommunications, No.9 Wenyuan Road, Nanjing, Jiangsu, 210023, People's Republic of China

[2] School of Computing and Information Sciences, Florida International University, 11200 SW 8th Street, Miami, FL, 33199, USA

representative examples of micro-blogging service providers, allows users to post short messages, *tweets*, within 140-character limit. One particular topic Twitter that users publish tweets about is "what's happening", which makes Twitter differentiated from news media with its real-time nature [24, 31, 40, 47]. For example, we could detect a tweet related to a shooting crime 10 minutes after shots fired, while the first new report appeared approximately three hours later. Meanwhile, tweets have a broad coverage over all types of real-world events, accounting for Twitter's large number of users, including verified accounts such as news agents, organizations and public figures. The real-time event information is particularly useful for keep people informed and updated on the events happening in real-world with their user-contributed messages.

On the other hand, the large number of messages from millions of social media users often leads to the information overload problem [27, 42, 48, 49]. Although the large volume of tweets provides enough information about events, because of a lot of noises, it is not straightforward and sometimes quite difficult for people themselves to access the real information about a particular event from the Twitter stream. In addition, those who search for the information related to a particular event often find it is difficult to quickly obtain an overview of the events, given the overwhelmingly large collection of data. Thus, to make use of Twitter's real-time nature, it is imperative to develop effective automatic methods to conduct event detection, detecting events from a Twitter stream by identifying important moments in the stream and their associated tweets [11, 12, 17].

Social event summarization aims to provide a textual description of an event of interest to address the aforementioned problem [7, 20, 46]. Given a data stream consisting of chronologically-ordered text pieces related to an event, a social event summarization system aims to generate an informative textual description that can capture all the important moments. Ideally the summary should be produced in a progressive manner as the event unfolds.

1.2 Content of the paper

In this paper, we present a participant-centered social event summarization framework. The proposed framework consists of three key components: participant detection, sub-event detection, and summary tweet extraction. Instead of detecting important moments directly, we first dynamically identify participants, which are named entities frequently mentioned in the input stream, then "zooms-in" the whole stream to the participant level. To detect important moments related to each participant, we propose a time-content mixture model considering both volume changes and topic changes along the stream, so that associated posts of an event are not only temporally bursty but also topically coherent. Among different real-life events, the sports games receive a lot of attention from the Twitter audience. To evaluate the proposed framework, we use sports games as an application. The application provides an alternative way to be kept informed of the progress of a sports game as well as audience's responses from the social media data. Based on the event detection results, summarization are employed to summarize the game's progress and audience's supports for different levels: an event, a participant and the whole game. The summary of the progress of a game can be delivered in real-time to the sports fans who cannot watch the game; the automatically generated summary can also be supplied to the news reporters to assist with the writing of the game recap (e.g., providing a full coverage of the exciting moments happened on the playground). To make the system applicable in real data, an online clustering approach is developed for participant detection and an online temporal-content mixture model is proposed to conduct sub-event detection. Experiments show that the proposed framework can achieve similar performance with its batch counterpart.

The rest of the paper is organized as follows: Section 2 reviews related work. Section 3 gives an overview of the proposed participant-based social event summarization framework. Section 4 formulates the participant detection problem and develops an online clustering approach for online participant detection. Section 5 proposes an online temporal-content mixture model to conduct sub-event detection. Section 6 presents the experimental results. Finally Section 7 concludes the paper and discusses future work.

2 Related work

In this section, we divide the related studies into two categories: document summarization and event detection.

2.1 Multi-document summarization

As a fundamental and effective tool for document understanding and organization, multi-document summarization enables better information service by creating concise and informative reports (i.e., a generated summary) for a large collection of documents [21]. The generated summary can be generic where it simply gives the important information contained in the input documents without any particular information needs or query/topic-focused where it is produced in response to a user query or related to a topic [21, 26].

For the last over two decades, multi-document summarization has attracted attention of a large number of researchers, and various aspects of the problem have been explored and many methods have been proposed. Typical methods include Maximal Marginal Relevance(MMR) [14], centroid-based methods [29, 34], graph-based methods [10, 44], rank-based methods [10, 44], matrix-factorization based methods [43], probability models [9, 18, 38], and topic models [41].

In this paper, based on the event detection results, document summarization is applied to the multiple tweets of each sub-event to summarize the game's progress and audience's supports for different levels: an event, a participant, and the whole game.

2.2 Event detection

The concept of event detection is first introduced by Topic detection and tracking (TDT), which is a research program initiated by DARPA (Defense Advanced Research Projects Agency) for finding and following the new events in streams of broadcast news stories.[1] Attempts have been made to adapt the methods developed on formal document collections to event detection on social media, including even new event detection from a stream of Twitter posts based on locality-sensitive hashing [32], online clustering [5], data collection and event recognition [50], bursty topic detection [8], open-domain event-extraction and categorization [35], visualization [27], and summarization [6, 28, 48].

In this paper, we propose a participant-based method to detect important moments along a social media stream. To detect important moments related to each participant, we propose a time-content mixture model considering both volume changes and topic changes along the stream and important moments detected for different participants.

[1] http://projects.ldc.upenn.edu/TDT/

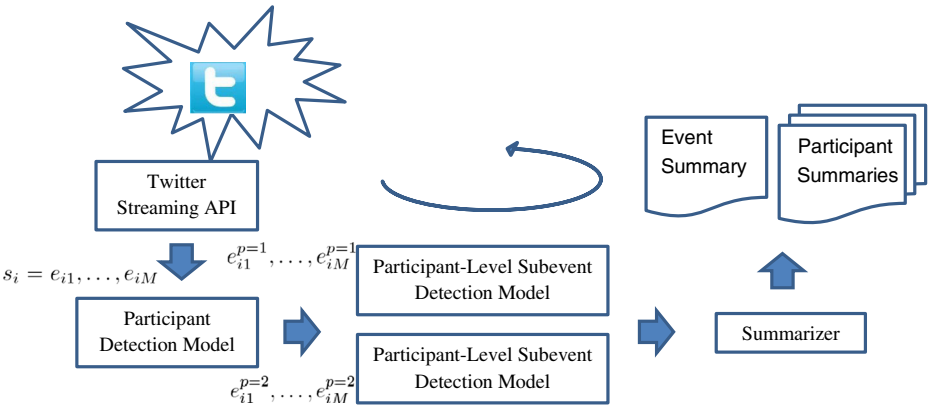

Figure 1 System framework of the event summarization application for sports games using Twitter streams

3 The framework overview

We propose a novel participant-centered event summarization approach that consists of three key components: (1) "Participant Detection" dynamically identifies the event participants and divides the entire event stream into a number of participant streams; (2) "Sub-event Detection" introduces a novel time-content mixture model approach to identify the important sub-events associated with each participant; these "participant-level sub-events" are then merged along the timeline to form a set of "global sub-events",[2] which capture all the important moments in the event stream; (3) "Summary Tweet Extraction" extracts the representative tweets from the global sub-events and forms a comprehensive coverage of the event progress.

In Figure 1, we provide an overview of the system framework. It consists of three main components: participant detection, sub-event detection, and summary generation.

To collect the stream of tweets about a particular event, the system requires users to input the start and end time of the event, and a set of keywords, and calls Twitter's streaming APIs the obtain tweets containing one of the keywords during the event's time period. The three key components are described below.

- **Participant Detection**: The goal of participant detection is to identify the important entities in the stream that play a significant role in shaping the event progress [37]. We introduce an online clustering approach to automatically group the mentions referred to the same entities in the stream, and update the model for every input segment of tweets s_i. According to the clustering results, the input segment can be devided into several sub-segments, one for each participant p, as s_i^p, composed of those tweets of s_i containing a mention of the participant p.
- **Sub-event Detection**: Given a participant stream, the proposed sub-event detection algorithm automatically identify the important moments (a.k.a. sub-events) in the

[2]We use "**participant sub-events**" and "**global sub-events**" respectively to represent the important moments happened on the participant-level and on the entire event-level. A "global sub-event" may consist of one or more "participant sub-events". For example., the "steal" action in the basketball game typically involves both the defensive and offensive players, and can be generated by merging the two participant-level sub-events.

stream based on both content salience and the temporal burstiness of the stream. Each sub-event is represented by a set of associated tweets and a peak time, when the tweet volume has reached a peak during that time period.

- **Summary Generation**: The summary generation module takes the input of sets of tweets, each associated with a sub-events of a participant, and aims to generate a high-quality textual summary as well as a sentiment summary.

To build the application for sports games, we first obtain a filtered Twitter stream using a set of keywords including names of teams, players and coaches. Then the participant-based sub-event detection is applied on the event stream data to detect the important moments (sub-events) during the event. The dominating set based summarization approach is then applied to the multiple tweets of each sub-event [13, 36].

Figures 2 and 3 show the screen-shots of our system. In Figure 2, users can choose to replay a previous event or follow a current ongoing event. As the related tweets of the chosen event are being fed into the system, filtered by predefined keywords related with the event, new sub-events are detected and summarized automatically, and inserted into the top part of the main section of the page. The right side of the page lists participants of the event. The number by each participant indicates the number of tweets where this participant is discussed, by which users can find the most popular participants so far. To obtain more information about a participant that users are interested in, they can further zoom-in to a particular participant to list all the sub-events the participant is involved in so far. After users click the arrow icon beside a sub-event summary in Figure 2, a detailed page of the sub-event is then shown in Figure 3, including the list of all tweets about the sub-events and a sentimental analysis result. For showing the aggregated sentiment of Twitter users for

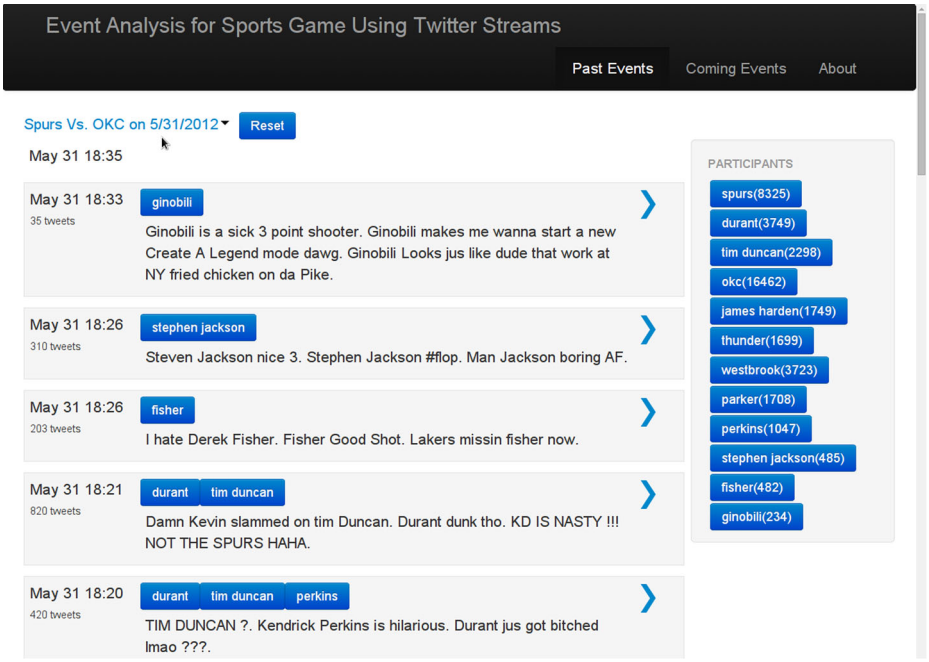

Figure 2 Screenshot of the sub-event list of the system

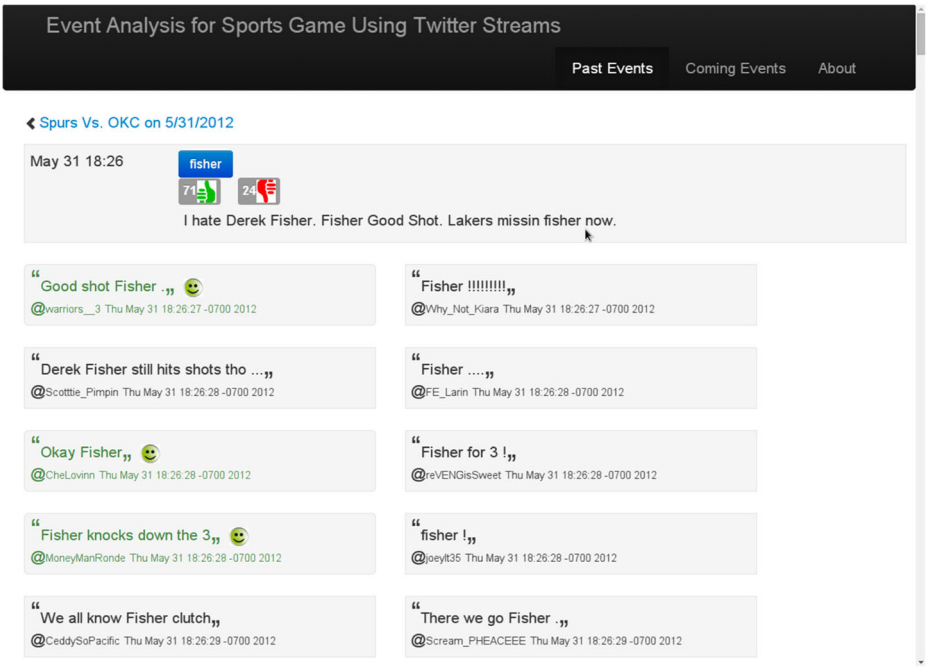

Figure 3 Screenshot of the sub-event details of the system

each sub-event, the system calculates the numbers of positive and negative tweets of the sub-event respectively, after conducting a sentiment classification on each tweet.

4 Online participant detection

4.1 Introduction

We define event participants as the entities that play a significant role in the event. "Participant" is a general concept to denote the event participating persons, organizations, product lines, etc., each of which can be captured by a set of correlated proper nouns. For example, the NBA player "*LeBron Raymone James*" can be represented by {*LeBron James, LeBron, LBJ, King James, L. James*}, where each proper noun represents a unique mention of the participant [3, 30, 40]. In this work, we automatically identify the proper nouns from tweet streams, filter out the infrequent ones using a threshold ψ, and cluster them into individual event participants. This process allows us to dynamically identify the key participating entities and provide a full-coverage for these participants in the detected events.

Most of existing approaches [27, 42, 48, 49] rely on changes of tweet volumes by detecting bursts in the stream as important moments, and assume all tweets during a burst describe the corresponding event. However in real cases, because of average effects of multiple topics existing in the stream, important moments, in term of one topic, which may lead bursts among posts about the topic, may not be well reflected in changes of post volumes in the whole stream. Using participant-based approach, the important events related to each participant can be easily identified. This can be demonstrated using an example in Figure 4, in

Figure 4 Example Twitter event stream (*upper*) and participant stream (*lower*)

which upper subfigure is a Twitter stream which is composed of tweets related to a NBA game Spurs vs Thunder, and the lower subfigure is its sub-stream which contains only tweets corresponding to the player Russell Westbrook in this game.

4.2 Participant detection

For the online requirement, we formulate the participant detection as an incremental cross-tweets co-reference resolution task in a twitter stream. A named entity recognition tool [33] is used for named entity tagging in tweets. Then the tagged named entities (a.k.a., mentions) are grouped into clusters using a streaming clustering algorithm, which consists of two stages: the update stage and the merge stage, applied to each new incoming segment of tweets. The update stage adds mentions to existing clusters if the similarity between the mention and an existing cluster is less than a threshold δ_u or otherwise creates new clusters, while the merge stage itself is hierarchical agglomerative clustering to revise the clustering result by combining them.

In the update stage, we define the similarity of a mention m and an existing cluster c as

$$sim(m, c) = \alpha lex(m, c) + (1 - \alpha) context(m, c), \tag{1}$$

where $lex(m, c)$ captures lexical resemblance between m and mentions in c and and $context(m, c)$ cosine similarity between contexts of m and c. $lex(m, c)$ can be calculated as portion of overlapping n-grams between them as

$$lex(m, c) = \frac{|ngram(m) \cap ngram(c)|}{|ngram(m) \cup ngram(c)|}. \tag{2}$$

For example, in the following two tweets "*Gotta respect Anthony Davis, still rocking the unibrow*", "*Anthony gotta do something about that unibrow*", the two mentions *Anthony*

Davis and *Anthony* are referring to the same participant and they share both character overlap ("anthony") and context words ("unibrow", "gotta"). However, for mentions in tweets, their context information is very limited and may vary a lot even they referred to the same entity. The previous update process may lead to a large of number of new clusters which lower efficiency of the system. Instead of updating the clustering by one mention each time, by assuming that mentions in one segment with the same name refer to the same entity, we first group all mentions with the same name in a segment, extract context for the mentions and select a cluster to assign all these mentions to.

To further reduce the cluster number, since participants we want to detect are entities that play significant roles, we can discard some infrequent entities. For a name, if there are more than δ_l continuous slices in each of which there are more than δ_s mentions, we then activate the name. So we only keep track of mentions with frequent names.

In the merge stage, a hierarchical agglomerative clustering is conducted with a stopping threshold δ_m. Since we suppose to have sufficient context information in this stage and our goal to combine mentions with different names, here only the context similarity is used to measure the similarity between clusters while lexical resemblance is used as constraints. To combine two clusters, at least half of mentions in both clusters needs to be lexically related with a mention in each other. A mention m is lexically related with mention m' if $m(m')$ is an abbreviation, acronym, or part of another mention $m'(m)$, or or if the character edit distance between the two mentions is less than a threshold θ. In our study, θ was empirically set as $0.2 \times min\{|m|, |m'|\}$.

5 Online update for a temporal-content mixture model

When we have all the tweets about the event, the EM algorithm can be applied to the whole data to train the event detection model. In this paper, we first present a basic *Temporal-Content Mixture Model* for static data analysis. Then an online algorithm, which is designed for summarizing Twitter's on-going events in real-time, is presented.

5.1 Temporal-content mixture model

An event corresponds to a topic that emerges from the data stream, being intensively discussed during a time period, and then gradually fades away [37]. The tweets corresponding to an event thus demand not only "temporal burstiness" but also a certain degree of "lexical cohesiveness". To incorporate both the time and content aspects of the events, we use a mixture model approach proposed in [37] for event detection. Figure 5 shows the plate notation.

In the proposed mixture model, each tweet d in the data stream D is generated from a topic z, weighted by π_z. Each topic is characterized by both its content and time aspects. The content aspect is captured by a multinomial distribution over the words, parameterized by θ; while the time aspect is characterized by a Gaussian distribution, parameterized by μ and σ, with μ represents the average time point that the event emerges and σ determines the duration of the event. These distributions bear similarities with the previous work [1, 15, 18]. In addition, there are often background or "noise" topics that are being constantly discussed over the entire event evolvement process and do not present the desired "burstiness" property. We use a uniform distribution $U(t_b, t_e)$ to model the time aspect of these "background" topics, with t_b and t_e being the event beginning and end time points. The content aspect of a background topic is modeled by similar multinomial distribution, parameterized

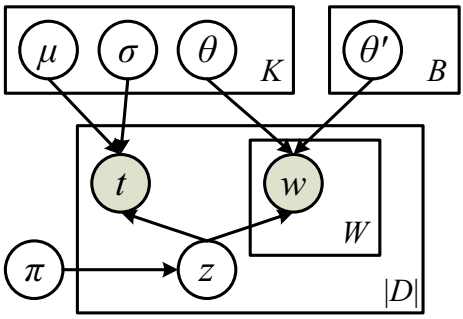

Figure 5 Plate notation of the mixture model

by θ'. We use the maximum likelihood parameter estimation. The data likelihood can be represented as:

$$L(D) = \prod_{d \in D} \sum_{z} \{\pi_z p_z(t_d) \prod_{w \in d} p_z(w)\}$$

where $p_z(t_d)$ models the timestamp of tweet d under the topic z; $p_z(w)$ corresponds to the word distribution in topic z. They are defined as:

$$p_z(t_d) = \begin{cases} N(t_d; \mu_z, \sigma_z) & \text{if } z \text{ is an event topic} \\ U(t_b, t_e) & \text{if } z \text{ is background topic} \end{cases}$$

$$p_z(w) = \begin{cases} p(w; \theta_z) & \text{if } z \text{ is an event topic} \\ p(w; \theta'_z) & \text{if } z \text{ is background topic} \end{cases}$$

where both $p(w; \theta_z)$ and $p(w; \theta'_z)$ are multinomial distributions over the words. Initially, we assume there are K event topics and B background topics and use the EM algorithm for model fitting. The EM equations are listed below:

E-step:

$$p(z_d = j) \propto \begin{cases} \pi_j N(d; \mu_j, \sigma_j) \prod_{w \in d} p(w; \theta_j) & \text{if } j <= K \\ \pi_j U(t_b, t_e) \prod_{w \in d} p(w; \theta'_j) & \text{else} \end{cases}$$

M-step:

$$\pi_j \propto \sum_d p(z_d = j)$$

$$p(w; \theta_j) \propto \sum_d p(z_d = j) \times c(w, d)$$

$$p(w; \theta'_j) \propto \sum_d p(z_d = j) \times c(w, d)$$

$$\mu_j = \frac{\sum_d p(z_d = j) \times t_d}{\sum_{j=1}^{K} \sum_d p(z_d = j)}$$

$$\sigma_j^2 = \frac{\sum_d p(z_d = j) \times (t_d - \mu_j)^2}{\sum_{j=1}^{K} \sum_d p(z_d = j)}$$

During EM initialization, the number of event topics K was empirically decided by scanning through the data stream and examine tweets in every 3-minute stream segment. If there was

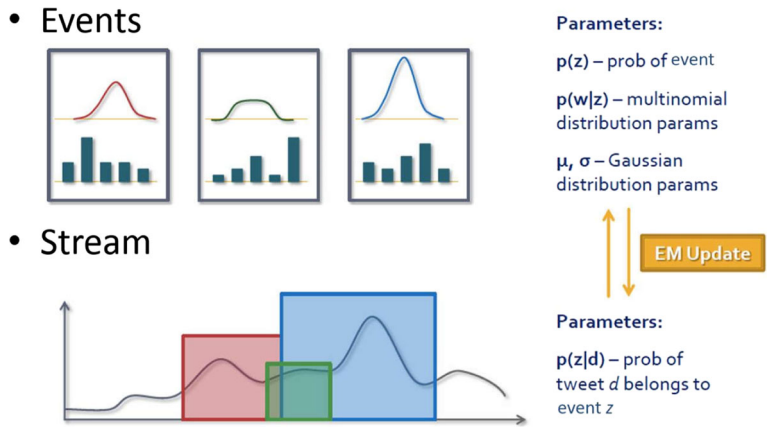

Figure 6 An illustration of time-content mixture model

a spike,[3] we add a new event to the model and use the tweets in this segment to initialize the value of μ, σ, and θ. Initially, we use a fixed number of background topics with $B = 4$. A topic re-adjustment was performed after the EM process. We merge two events in a data stream if they (1) locate closely in the timeline, with peaks times within a 2-minute window; and (2) share similar word distributions: among the top-10 words with highest probability in the word distributions, there are over 5 words overlap. We also convert the event topics to background topics if their σ values are greater than a threshold β.[4] We then re-run the EM process to obtain the updated parameters. The topic re-adjustment process continues until the number of events and background topics do not change further. An illustration of the time-content mixture model and the EM algorithm is shown in Figure 6.

5.2 Online update

In an online processing mode using the same temporal-content mixture model, the system iteratively consumes the new w_{new} slices of tweets each time to update the model parameters with the most recent $w_{working}$ slices of tweets in memory. The $w_{working}$ slices can be further divided into updating area, fixed area in Figure 7, where a Gaussian distribution is used to represent a sub-event topic.

Due to the locality of a sub-event, we assume independency between the sub-events before updating area (including reserved and fixed area) and the incoming tweets, so that only parameters for those sub-event topics in the updating area are updated with new incoming tweets. For the same reason, the oldest tweets in the fixed area are least likely to belong to a much older sub-event topic, so we only need to keep the parameters of the sub-event topics in reserved area in memory. In the application, we set 10 min for width of the updating area, 15 min for width of the reserved area, and 5 min for the fixed area to keep tweets of 20 min in memory.

A data segment is represented as w slices: $D_i = s_i, s_{i+1}, \ldots, s_{i+w-1}$. We use K and B to denote the number of sub-event topics and background topics currently contained in the

[3] We use the algorithm described in [27] as a baseline and ad hoc spike detection algorithm.

[4] β was set to 5 minutes in our experiments.

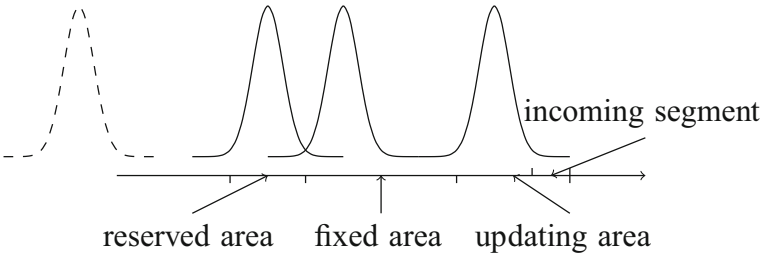

Figure 7 Illustration of how sub-events are detected online

model. B was empirically set to 2 initially. The following steps are repeated to process each data segment:

EM Initialization When a new data segment D_i becomes available, we need to update the number of sub-event topics ΔK and a background topic ΔB, as well as re-initialize the model parameters (μ, σ, θ) for both sub-event and background topics. Initially we set the increment of the sub-event topics empirically ($\Delta K = 1$) and keep the number of background topics unchanged ($\Delta B = 0$). Later we will perform a topic readjustment process to further adjust their numbers. For the new sub-event topics, its Gaussian parameters μ and σ are initialized using the tweets in the new data segment; its multinomial parameters are initialized randomly. The new data segment D_i also introduces unseen words which we use to expand our existing vocabulary. For both existing sub-event topics and background topics, the multinomial parameters corresponding to these new words are initiated randomly to a small value.

EM Update To perform the EM update, we only involve the sub-event topics that are most close to the current time point in the new EM update process. They are the ones whose peak time $\hat{t}$ is within updating area. Their parameters will likely be changed given a new segment of the data stream. The parameters of the earlier sub-event topics are fixed and will not be changed anymore. In addition, we would like to involve only the most recent tweets in the model update. We use only those tweets who are published in fixed area and updating area. Those tweets that are published earlier are discarded. These tweets are used together with the new data segment for the new EM update.

EM Postprocessing A topic re-adjustment was performed after the EM process. We merge two sub-events in a data stream if they (1) locate closely in the timeline, with peaks times within a 2-minute window, where peak time of a sub-event is defined as the slice that has the most tweets associated with this sub-event; and (2) share similar word distributions if their symmetric KL divergence is less than a threshold ($thresh_{sim} = 5$). We also convert the sub-event topics to background topics if their σ values are greater than a threshold β.[5] We then re-run the EM process to obtain the updated parameters. The topic re-adjustment process continues until the number of sub-events and background topics do not change further. We only output the sub-event topic is the number of associated tweets in its peak time is larger than a threshold (=15).

[5] β was set to 5 minutes in our experiments.

Table 1 Statistics of the data set, including five NBA basketball games event

Event		Date	Duration	#Tweets
	Lakers vs Okc	05/19/2012	3h10m	218,313
N	Celtics vs 76ers	05/23/2012	3h30m	245,734
B	Celtics vs Heat	05/30/2012	3h30m	345,335
A	Spurs vs Okc	05/31/2012	3h	254,670
	Heat vs Okc	06/21/2012	3h30m	332,223

5.3 Event Detection

We obtain the "**participant sub-events**" by applying this sub-event detection approach to each of the participant streams. The "**global sub-events**" are obtained by merging the participant sub-events along the timeline. We merge two participant sub-events into a global sub-event if (1) their peaks are within a 2-minute window, and (2) the Jaccard similarity [22] between their associated tweets is greater than a threshold (set to 0.1 empirically). The tweets associated with each global sub-event are the ones with $p(z|d)$ greater than a threshold γ, where z is one of the participant sub-events and γ was set to 0.7 empirically. After the sub-event detection process, we obtain a set of global sub-events and their associated event tweets. Note that we empirically set some threshold values in the topic re-adjustment and sub-event merging process. In the future, we would like to explore more principled way of parameter selection.

Once we have sets of tweets, each associated with a sub-events of a participant, we can utilize document summarization approaches to generate to high-quality textual summary. In particular, to build the application for sports games in our framework, the dominating set based summarization approach is then applied to the multiple tweets of each sub-event [13, 36].

6 Experiments

In this section, we evaluate our proposed summarization framework on five NBA basketball games. In particular, we use the heterogeneous event topics to verify that the proposed approach can robustly and efficiently detect events on different types of Twitter streams. The tweet streams corresponding to these topics are collected using the Twitter Streaming API[6] with pre-defined keyword set. For NBA games, we use the team names, first name and last name of the players and head coaches as keywords for retrieving the tweets related to the event topic;We crawl the tweets in real-time when these scheduled events are taking place; nevertheless, certain non-event tweets could be mis-included due to the broad coverage of the used keywords. During preprocessing, we filter out the tweets containing URLs, non-English tweets, and retweets since they are less likely containing new information regarding the event progress. Table 1 shows statistics of the event tweets after the filtering process. In total, there are over 1.8 million tweets used in the event detection experiments.

[6] https://dev.twitter.com/docs/streaming-apis

Table 2 An example clip of the play-by-play live coverage of an NBA game (Heat vs Okc)

Time	Action (Event)	Score
9:22	Chris Bosh misses 10-foot two point shot	7–2
9:22	Serge Ibaka defensive rebound	7–2
9:11	Kevin Durant makes 15-foot two point shot	9–2
8:55	Serge Ibaka shooting foul (Shane Battier draws the foul)	9–2
8:55	Shane Battier misses free throw 1 of 2	9–2
8:55	Miami offensive team rebound	9–2
8:55	Shane Battier makes free throw 2 of 2	9–3

We use the play-by-play live coverage collected from the ESPN[7] websites as reference, which provide detailed descriptions of the NBA as they unfold. Table 2 shows an example clip of the play-by-play descriptions of an NBA game, where "Time" corresponds to the minutes left in the current quarter of the game, and "Score" shows the score between the two teams. Ideally, each item in the live coverage descriptions may correspond to an event in the tweet streams, but in reality, not all actions would attract enough attention from the Twitter audience. We use a human annotator to manually filter out the actions that did not lead to any spike in the corresponding participant stream. The rest items are projected to the participant and event streams as the goldstandard events. The projection was manually performed since the "game clock" associated with the goldstandard (first column in Table 2) does not align well with the "wall clock" due to the game rules such as timeout and halftime rest.

6.1 Participant detection

We evaluate the participant detection similar as a cross-tweet co-reference solution task. To build labeled co-reference data, for every event, we first sample hundreds to over a thousand tweets containing one of 50 most frequent names in the event; then an annotator labeled these sampled tweets with chains of entities. Singletons and those mentions which are not referred to an actually participant of the event (e.g., "Kevin" referred to a cousin of the tweet author, or "Jessica" referred to a performer on American Idols). B-Cubed [4], is most widely used in co-reference resolution evaluation, is used as the metric compare participant detection result and the labeled data. Recall score of B-Cubed is calculated as:

$$B_R^3 = \frac{1}{N} \sum_{d \in D} \sum_{m \in d} \frac{O_m}{S_m} \tag{3}$$

where D, d and m are the set of documents, a document, and a mention, respectively. S_m is the set of mentions of the annotated mention chain which contains m, while O_m is the overlap of S_m and the set of mentions of the system generated mention chain which contains m. N is the total number of mentions in D. The precision is computed by switching the role of annotated data and system generated data. F-measure is computed as geometrical average of recall and precision.

[7] http://espn.go.com/nba/scoreboard

We evaluate the participant detection method used in the application system, referred to as SegmentUpdate, by comparing it with following baselines:

- **ExactMatch:** The method clusters mentions only based on names.
- **TweetUpdate:** In update stage, clustering is updated once for a mention in a tweet.
- **IncNameHAC:** It is an incremental version of NameHAC, updating the hierarchical tree based on the available part of the stream, by conducting further merge.
- **NameHAC:** Hierarchical agglomerative clustering on names of mentions, assuming mentions with the same name refer to the same entity. For a pair of names, their similarity is based on the whole stream, so it is not applicable to our case, but can be seen as an upper bound.

Table 3 shows the comparing results. We can observe that 1) NameHAC has the best performance since it makes use of the whole data instead of conducting detection incrementally; 2) The incremental version of NameHAC does not perform well, even worse than the trivial method ExactMatch; 3) SegmentUpdate, which is used the application system, has a reasonable performance. It outperforms IncNameHAC since it allow two mentions composed of the same phrase refer to different participants, if the phrase is ambiguous. It also performs better than TweetUpdate, since it collects more information in phrase clustering for each phrase, from a segment of tweets instead of a single tweet.

6.2 Event detection results

We compare our proposed time-content mixture model (noted as "MM") against the spike detection algorithm proposed in [27] (noted as "Spike"). The spike algorithm is based on the tweet volume change. It uses 10 seconds as a time unit, calculates the tweet arrival rate in each unit, and identifies the rates that are significantly higher than the mean tweet rate. For these rate spikes, the algorithm finds the local maximum of tweet rate and identify a window surrounding the local maximum. We tune the parameter of the "Spike" approach

Table 3 Performance comparison of methods for participant detection

Approach	Lakers vs Okc			Celtics Vs 76ers			Celtics vs Heat		
	P	R	F	P	R	F	P	R	F
ExactMatch	0.981	0.692	0.811	0.825	0.585	0.685	0.893	0.696	0.782
TweetUpdate	1.000	0.658	0.794	0.913	0.660	0.766	0.847	0.720	0.779
IncNameHAC	1.000	0.542	0.703	0.820	0.589	0.686	0.822	0.650	0.726
SegmentUpdate	1.000	0.682	0.811	0.851	0.707	0.772	0.801	0.855	0.827
NameHAC	1.000	0.791	0.883	0.875	0.716	0.788	0.8884	0.918	0.903
	spursvsokc			heatvsokc					
	P	R	F	P	R	F			
ExactMatch	0.857	0.616	0.717	0.922	0.626	0.746			
TweetUpdate	0.877	0.712	0.786	0.952	0.712	0.815			
IncNameHAC	0.864	0.545	0.669	0.932	0.753	0.833			
SegmentUpdate	0.839	0.764	0.800	0.911	0.847	0.878			
NameHAC	0.853	0.774	0.811	0.948	0.843	0.892			

Table 4 Event detection results on participant streams

Event	**Participant-level Event Detection**							
	#P	#S	Spike			MM		
			R	P	F	R	P	F
Lakers vs Okc	9	65	0.75	0.31	**0.44**	0.71	0.39	**0.50**
Celtics vs 76ers	10	88	0.52	0.39	**0.45**	0.53	0.43	**0.47**
Celtics vs Heat	14	152	0.53	0.29	**0.37**	0.50	0.38	**0.43**
Spurs vs Okc	12	98	0.78	0.46	**0.58**	0.84	0.57	**0.68**
Heat vs Okc (1)	15	123	0.75	0.27	**0.40**	0.72	0.35	**0.47**
Heat vs okc (2)	13	153	0.74	0.36	**0.48**	0.76	0.43	**0.55**
WWDC'12	10	56	0.64	0.14	**0.23**	0.59	0.33	**0.42**
Average	**12**	**105**	0.67	0.32	**0.42**	0.66	0.41	**0.50**

(set $\tau = 4$) so that it yields similar recall values as the mixture model approach. We then apply the "MM" and "Spike" approaches to both the participant and event streams and evaluate the event detection performance. Results are shown in Table 4. A system detected event is considered to match the goldstandard event if its peak time is within a 2-minute window of the goldstandard.

We apply the "Spike" and "MM" approach to the participant streams. The participant streams on which we cannot detect any meaningful events have been excluded, the resulting number of participants are listed in Table 4 and denoted as "#P", and "#S" is the summation number of events from all participant streams of each input dataset. In general, we found the "MM" approach can perform better since it inherently incorporates both the "burstiness" and "lexical cohesiveness" of the event tweets, while the "Spike" approach relies solely on the "burstiness" property. Note that although we divide the entire event stream into participant streams, some key participants still own huge amount of discussion and the spike patterns are not always clearly identifiable. The time-content mixture model gains advantages in these cases.

6.3 Event summarization

In this section, we evaluate the performance of our proposed summarization framework. For each game, an annotator manually labels the sub-events according the play-by-play data from ESPN,[8] and for each sub-event, representative tweets are extracted up to 140 characters as the manual summary.

To evaluate the final summaries of an event, we following the work in [39] to evaluate summarization for a document stream using a modified version of ROUGE [23] score, which widely used as automatic evaluation for document summarization tasks. ROUGE measures the quality of a summary by counting the unit overlaps between the candidate summary and a set of reference summaries. Several automatic evaluation methods are

[8] http://espn.go.com/nba/scoreboard

implemented in ROUGE, such as ROUGE-N, ROUGE-L, ROUGE-W and ROUGE-SU. ROUGE-N is an n-gram recall computed as follows:

$$\text{ROUGE-N} = \frac{\sum_{S \in \text{ref}} \sum_{\text{gram}_n \in S} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \text{ref}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)}, \tag{4}$$

where n is the length of the n-gram, ref stands for the reference summaries, $\text{Count}_{\text{match}}(\text{gram}_n)$ is the number of co-occurring n-grams in a candidate summary and the reference summaries, and $\text{Count}(\text{gram}_n)$ is the number of n-grams in the reference summaries. ROUGE-L uses the longest common sub-sequence (LCS) statistics, while ROUGE-W is based on weighted LCS and ROUGE-SU is based on skip-bigram plus unigram. Each of these evaluation methods in ROUGE can generate three scores (recall, precision and F-measure). However, ROUGE score cannot be applied directly to summarization of a document stream, in our case, a tweet stream about an event, since same n-grams that appear at distant time points describe different sub-events and should be regarded as different n-grams. In our manually labeled and system generated summaries, each n-gram is associated with the timestamp as the same of the sub-event the n-gram describes. Making use of such temporal information, we modify ROUGE-N to ROUGE^T-N, calculated as

$$\text{ROUGE}^T\text{-N} = \frac{\sum_{S \in \text{ref}} \sum_{\text{gram}_n^t \in S} \text{Count}_{\text{match}^T}(\text{gram}_n^t)}{\sum_{S \in \text{ref}} \sum_{\text{gram}_n^t \in S} \text{Count}(\text{gram}_n^t)}, \tag{5}$$

where gram_n^t is a unique n-gram with a timestamp, and $\text{Count}_{\text{match}^T}(\text{gram}_n^t)$ returns the minimum of occurrence of n-gram with timestamp t in S and the number of matched n-grams in a candidate summary. The distance between the timestamp of a matched n-gram and t needs to be within a constant, which set to 1 min in our experiments.

We compare the sub-event detection method used in the application system, referred to as MixtureModelOnline+Participant to the spike detection method (Spike) [27] and the method batch-mode (MM) with or without participant detection results. Table 5 shows the summarization evaluation results for comparing sub-event detection methods in term of the new evaluation metric $\text{ROUGE}^{\text{T}} - 1$ F-1 score. From Table 5, we have several observations: 1) sub-event detection conducted based on participant streams leads to better summarization performance due to more accurate sub-event detection results; 2) The temporal-content mixture model outperforms the spike detection since the former takes the tweet content into consideration; 3) The online version of temporal-content mixture model, MMOnline+Participant, under-performs its batch counterpart, but their F-1 scores are close, which indicates that it still can lead to a reasonable performance in the real application system.

Table 5 ROUGE^{T}-1 F-1 scores

Methods	Celtics Vs 76ers	Celtics vs Heat	Heat Vs Okc	Lakers vs Okc	Spurs Vs Okc
Spike	.2664	.31651	.2736	.2838	.2409
+Participant	.3240	.38784	.3016	.3399	.2917
MM	.3199	.38591	.3286	.3526	.2841
+Participant	.3571	.40162	.3493	.3899	.3063
MMOnline +Participant	.3428	.3970	.3163	.3852	.3068

7 Conclusion

Social media data plays a more and more important role in our daily lives and in many real applications. In this paper, we present an event summarization application for sports games using Twitter streams, integrating the techniques. To make the system applicable in real data, we propose the online version of participant based temporal-content mixture model to conduct sub-event detection. Experiments show that it can achieve similar performance with its batch counterpart.

There are several avenues for future research: (1) Use of natural language processing techniques. Natural language processing (NLP) provides the fundamental basis for the upper layer of text analysis. There are many advanced NLP techniques can be used and adapted to the analysis of the social media data. (2) Integration of social network information: Current analysis tasks are generally based on the content of documents/posts. However, in social media, documents/posts contain not only content but also users information, which further composes the whole social network. So text analysis can utilize user profiles and user communities etc. In addition, other typical information of social networks like geotags, and organization structure like dialogs can be used to understand the documents/posts more accurately. (3) More Applications. Social media has a large impact in a wide range of applications including hotel reviews, disaster management, and identification recognition [25, 45]. We can extend our current work to various application domains to support the software development of applications based on analysis of social media data.

Acknowledgements The work was supported in part by the National Science Foundation under Grant Nos. IIS-1213026, CNS-1126619, and CNS-1461926, Chinese National Natural Science Foundation under grant 91646116, Ministry of Education/China Mobile joint research grant under Project No.5-10, and Scientific and Technological Support Project (Society) of Jiangsu Province (No. BE2016776).

References

1. Allan, J.: Topic Detection and Tracking: Event-based Information Organization. Kluwer Academic Publishers Norwell, MA, USA (2002)
2. Ahlqvist, T., Beck, A., Halonen, M., Heinonen, S.: Social Media Roadmaps: Exploring the futures triggered by social media. VTT Tiedotteita - Research Notes (2454) (2008)
3. Atefeh, F., Khreich, W.: A survey of techniques for event detection in twitter. Comput. Intell., 132–164 (2013)
4. Bagga, A., Baldwin, B.: Algorithms for Scoring Coreference Chains The First International Conference on Language Resources and Evaluation Workshop on Linguistics Coreference (1998)
5. Becker, H., Naaman, M., Gravano, L.: Beyond Trending Topics: Real-World Event Identification on Twitter Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media, pp. 438–441 (2011)
6. Chakrabarti, D., Punera, K.: Event Summarization Using Tweets Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media, pp. 66–73 (2011)
7. Chang, Y., Wang, X., Mei, Q., Liu, Y.: Towards Twitter Context Summarization with User Influence Models Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, pp. 527–536 (2013)
8. Diao, Q., Jiang, J., Zhu, F., Lim, E.: Finding Bursty Topics from Microblogs Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pp. 536–544 (2012)
9. Daumé, IIIH., Marcu, D.: Bayesian Query-Focused Summarization Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, pp. 305–312 (2006)
10. Erkan, G., Radev, D.R.: Lexrank: Graph-based lexical centrality as salience in text summarization. JAIR **22**(1), 457–479 (2004)

11. Guille, A., Favre, C.: Event detection, tracking, and visualization in Twitter: a mention-anomaly-based approach. Soc. Netw. Anal. Min., 18:1–18:18 (2015)
12. Goswami, A., Kumar, A.: A survey of event detection techniques in online social networks. Soc. Netw. Anal. Min., 107 (2016)
13. Guha, S., Khuller, S.: Approximation algorithms for connected dominating sets. Algorithmica **20**(4), 374–387 (1998)
14. Goldstein, J., Mittal, V., Carbonell, J., Kantrowitz, M.: Multi-Document Summarization by Sentence Extraction. In: NAACL-ANLP 2000 Workshop on Automatic Summarization, Association for Computational Linguistics, pp. 40–48 (2000)
15. Hofmann, T.: Probabilistic Latent Semantic Indexing. In: Proceedings of the 22th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 50–57 (1999)
16. Hong, L., Dom, B., Gurumurthy, S., Tsioutsiouliklis, K.: A Time-dependent Topic Model for Multiple Text Streams. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 832–840 (2011)
17. He, R., Liu, Y., Yu, G., Tang, J., Hu, Q., Dang, J.: Twitter summarization with social-temporal context. World Wide Web, 1–24 (2017)
18. Haghighi, A., Vanderwende, L.: Exploring Content Models for Multi-Document Summarization. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, pp. 362–370 (2009)
19. Imran, M., Castillo, C., Diaz, F., Vieweg, S.: Processing Social Media Messages in Mass Emergency: A Survey. ACM Comput. Surv., 67:1–67:38 (2015)
20. Inouye, D., Kalita, J.K.: Comparing Twitter Summarization Algorithms for Multiple Post Summaries. In: Proceedings of 2011 IEEE third International Conference on Social Computing, pp. 290–306 (2011)
21. Jurafsky, D., Martin, J.: Speech and language processing. Prentice Hall, New York (2008)
22. L.L.: Measures of Distributional Similarity. In: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics, pp. 25–32 (1999)
23. Lin, C.-Y.: ROUGE: a Package for Automatic Evaluation of Summaries. In: Text Summarization Branches Out: Proceedings of the ACL-04 Workshop, pp. 74–81 (2004)
24. Li, Z., Tang, J., Wang, X., Liu, J., Lu, H.: Multimedia News Summarization in Search, ACM Trans. Intell. Syst. Technol., 33:1–33:20 (2016)
25. Li, T., Xie, N., Zeng, C., Zhou, W., Zheng, L., Jiang, Y., Yang, Y., Ha, H.-Y., Xue, W., Huang, Y., Chen, S.-C., Navlakha, J., Iyengar, S.S.: Data-Driven Techniques in Disaster Information Management. ACM Comput. Surv. **50**(1), 1:1–1:45 (2017)
26. Mani, I.: Automatic summarization. Comput. Linguist. 28(2)
27. Marcus, A., Bernstein, M., Badar, O., Karger, D., Madden, S., Miller, R.: Twitinfo: Aggregating and Visualizing Microblogs for Event Exploration. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 227–236 (2011)
28. Nichols, J., Mahmud, J., Drews, C.: Summarizing Sporting Events Using Twitter. In: Proceedings of the 2012 ACM Interntional Conference on Intelligent User Interfaces, pp. 189–198 (2012)
29. Nenkova, A., Vanderwende, L.: The impact of frequency on summarization. Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005-101
30. Purushotham, S., Kuo, C.-C.J.: Personalized Group Recommender Systems for Location- and Event-Based Social Networks. ACM Trans. Web, 16:1–16:29 (2016)
31. Peng, B., Li, J., Chen, J., Han, X., Xu, R., Wong, K.F.: Trending Sentiment-Topic detection on twitter. Springer International Publishing, 66–77 (2015)
32. Petrovic, S., Osborne, M., Lavrenko, V.: Streaming First Story Detection with Application to Twitter. In: Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 181–189 (2010)
33. Ritter, A., Clark, S., Mausam, O.: Etzioni, Named Entity Recognition in Tweets: an Experimental Study. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1524–1534 (2011)
34. Radev, D., Jing, H., Styś, M., Tam, D.: Centroid-based summarization of multiple documents. Inf. Process. Manag. **40**(6), 919–938 (2004)
35. Ritter, A., Mausam, O., Etzioni, S.: Clark, Open Domain Event Extraction from Twitter. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1104–1112 (2012)
36. Shen, C., Li, T.: Multi-Document Summarization via the Minimum Dominating Set. In: Proceedings of the 23rd International Conference on Computational Linguistics, Association for Computational Linguistics, pp. 984–992 (2010)

37. Shen, C., Liu, F., Weng, F., Li, T.: A Participant-based Approach for Event Summarization Using Twitter Streams. In: Proceedings of NAACL-HLT, pp. 1152–1162 (2013)
38. Tang, J., Yao, L., Chen, D.: Multi-Topic Based Query-oriented Summarization. In: Proceedings of SDM, pp. 1147–1158 (2009)
39. Takamura, H., Yokono, H., Okumura, M.: Summarizing a Document Stream. In: Proceedings of the 33rd European Conference on Advances in Information Retrieval, pp. 177–188 (2011)
40. Unankard, S., Li, X., Sharaf, M.A.: Emerging event detection in social networks with location sensitivity. World Wide Web, 1393–1417 (2015)
41. Wan, X.: Topic Analysis for Topic-Focused Multi-Document Summarization. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, pp. 1609–1612. ACM (2009)
42. Weng, J., Lee, B.-S.: Event Detection in Twitter. In: Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media, pp. 401–408 (2011)
43. Wang, D., Li, T., Zhu, S., Ding, C.: Multi-Document Summarization via Sentence-Level Semantic Analysis and Symmetric Matrix Factorization. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 307–314. ACM (2008)
44. Wan, X., Yang, J., Xiao, J.: Towards an Iterative Reinforcement Approach for Simultaneous Document Summarization and Keyword Extraction. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, pp. 543–552 (2007)
45. Xue, W., Li, T., Rishe, N.: Aspect identification and ratings inference for hotel reviews. World Wide Web **20**(1), 23–37 (2017)
46. Yamamoto, Y., Shinozaki, T., Ikegami, Y., Tsuruta, S.: Context respectful counseling agent virtualized on the Web. World Wide Web, 1–24 (2015)
47. Zhang, X., Li, Z., Zhu, S., Liang, W.: Detecting Spam and Promoting Campaigns in Twitter. ACM Trans. Web, 4:1–4:28 (2016)
48. Zubiaga, A., Spina, D., Amigó, E., Gonzalo, J.: Towards Real-time Summarization of Scheduled Events from Twitter Streams. In: Proceedings of the 23Rd ACM Conference on Hypertext and Social Media, pp. 319–320 (2012)
49. Zhao, S., Zhong, L., Wickramasuriya, J., Vasudevan, V., LiKamWa, R., Rahmati, A.: SportSense: Real-Time Detection of NFL Game Events from Twitter. Technical Report TR0511-2012
50. Zhao, S., Zhong, L., Wickramasuriya, J., Vasudevan, V.: Human as Real-Time Sensors of Social and Physical Events: A Case Study of Twitter and Sports Games. Technical Report TR0620-2011, Rice University and Motorola Labs