Event Detection in Twitter Using Multi Timing Chained Windows

Article *in* Computing and Informatics · May 2021

DOI: 10.31577/cai_2020_6_1336

Reza Ravanmehr
Islamic Azad University Central Tehran Branch

Computing and Informatics, Vol. 39, 2020, 1336–1359, doi: 10.31577/cai_2020_6_1336

EVENT DETECTION IN TWITTER USING MULTI TIMING CHAINED WINDOWS

Mohammad Mahdi Mojiri, Reza Ravanmehr*

Department of Computer Engineering
Central Tehran Branch, Islamic Azad University
Tehran, Iran
e-mail: mojiry@gmail.com, r.ravanmehr@iauctb.ac.ir

Abstract. Twitter is a popular microblogging and social networking service. Twitter posts are continuously generated and well suited for knowledge discovery using different data mining techniques. We present a novel near real-time approach for processing tweets and detecting events. The proposed method, Multi Timing Chained Windows (MTCW), is independent of the language of the tweets. The MTCW defines several Timing Windows and links them to each other like a chain. Indeed, in this chain, the input of the larger window will be the output of the smaller previous one. Using MTCW, the events can be detected over a few minutes. To evaluate this idea, the required dataset has been collected using the Twitter API. The results of evaluations show the accuracy and the effectiveness of our approach compared with other state-of-the-art methods in the event detection in Twitter.

Keywords: Event detection, microblogging, twitter, timing windows, MTCW

1 INTRODUCTION

Nowadays, social networks are much used in everyday lives, and the users of these networks usually share a lot of information with others about their events and incidents, their opinions on various issues, and so on. Meanwhile, Twitter has an essential role due to the number of users and its specific structure. This microblog had 336 million monthly active users in the first quarter of 2018 who sent 500 million

[*] Corresponding author

tweets per day [1, 2]. The high number of posts on this social network and the extent of its users have attracted the interest of researchers. In recent years, this huge amount of data has provided a hotbed for data mining research to discover facts, trends, events, and even predict specific incidents [3, 4, 5, 6, 7].

The data streaming process, including social networks, produces millions of documents (image, text, audio, etc.) per hour. Since these documents arrive at high speed, there is not much time to process them, and they cannot be stored in memory, either. Therefore, it is very difficult to compare the current and previous documents and find any similarities or changes between them.

On social networks such as Twitter, data is produced in an unlimited and endless stream. Therefore, the process for storing and analyzing this inexhaustible stream of information should not be dependent on the hardware specification of the system, such as memory capacity. In other words, if the event detection algorithm is applied to tweets for two months or two days, its accuracy should not be reduced, as well as the limitations of hardware such as memory to prevent it [5]. So, we need to propose an approach to detect events employing the analysis of tweets while they are received and require no further storage and processing.

The research conducted by Petrovic et al. [8] and Osborne and Dredze [9] confirm the effectiveness of employing Twitter for an event detection system. However, there are at least two severe challenges of doing so. Firstly, not all data from Twitter are correct and may include spam and personal information without processing capability. In studies such as [10], news agencies are the source of valid information, but the posts of users in social networks are not limited to factual data and events. Indeed, many posts are spam and trivial statements. The majority of tweets are not true stories, some concern personal life, some conversation and friendly chats, and some spam. The second challenge is the message length limit on Twitter. Twitter does not allow users to post a text longer than 280 characters, which causes changes in the syntax of the language. For example, short sentences are written instead of full sentences, which makes it difficult to process the tweets.

Many event detection approaches take advantage of the keywords contained in the users' post.The Burst Keyword is a group of keywords that represents a trend or the occurrence of a new event. Certainly, the period of the occurrence of these Burst Keywords is also very important; for example, if a Burst Keyword is observed over a few minutes, it represents an event, but if it occurs over several days, it indicates a new trend. The main idea of this research is to use multiple Timing Windows simultaneously and link them together. In this method, the smaller Timing Windows removes the words in the next (larger) Timing Windows and sends the remaining words to that Timing Windows. By applying this method, the history of repetitive words will not affect the detection of a sudden increase in windows at present. Moreover, processing of incoming tweets as well as other calculations are performed only for the first window, and there will be no waste of time for preliminary calculations in other Windows.

In the proposed method, Multi Timing Chained Windows (MTCW), sudden events can be detected over a few minutes. The suggested method is also independent of the user language. It benefits from the constant time and space complexity for any number of tweets, as explained in the next sections.

In the following, the research fundamentals of this study are described. The related works in the field of event detection are discussed in Section 3. The proposed approach is described in Section 4. Next, experimental results and evaluation of MTCW are discussed employing the collected Twitter dataset. Then, our proposed approach is compared with other state-of-the-art methods in event detection on Twitter. Finally, we conclude the paper and provide future works.

2 FUNDAMENTALS OF RESEARCH

In this section, the foundations of this research, including concepts, objectives, and quality of service parameters in event detection, are described.

2.1 Concepts in Event Detection

In recent years, the detection of an event has been a very trending topic. The underlying assumption is that an increasing use of some related keywords shows that an event is happening. Event is an occurrence of interest among users of the social media to discuss a real-world event-associated topic, usually after the incident or even sometimes prediction of it [11]. In the domain of Topic Detection and Tracking (TDT) [12], an event is defined as "Something that happens at a specific time and place along with all necessary conditions and unavoidable consequences".

In fact, there is no distinct segregation between trending topic detection and event detection in many papers. In some papers discussing the subject of trending topic detection, the term event detection is frequently mentioned, including [5, 13, 14, 15, 16, 17]. It is also pretty much the same in Twitter. In Twitter, there is a proprietary algorithm to detect and display the trending topics, consisting of terms and phrases that express the "trending" behavior. While Twitter's trending topics sometimes reflect current events (e.g., "iPhone X announcement"), they often include keywords for popular conversation topics (e.g., cryptocurrency).

As another example, suppose a celebrity who asks the fans in a tweet to comment on purchasing an apartment or a villa. A large number of tweets are generated discussing the disadvantages and benefits of each option. In this case, the "house" or "apartment" is raised as a trending topic, but has any event occurred in fact?

Indeed, the current systems usually are not intelligent enough to be able to recognize the difference between an event or a widespread discussion about a topic; therefore, most papers do not distinguish between an event and a trending topic. However, in articles such as [4], it has been attempted to use temporal and local tags to differentiate between urgent events with trends. It should be noted that Becker, Naaman, and Gravano [18] were the first researchers who considered this distinction between a trending topic and event detection.

2.2 Objectives of Event Detection

Considering the concepts and requirements of an event detection system, four significant objectives can be specified for an ideal user event detection system, including Generality, Scalability, Real-time processing, and No supervision. Apart from these four objectives, there are other criteria for perfect event detection. For example, no use of additional data and information or use of a dictionary could be good targets.

Generality: Many systems of event detection can only detect specific topics. In fact, just a series of topics already given to the system are detected. For example, an approach has been presented in [4] which detects whether the event is urgent or not and then classifies the event from among the preset categories (flood, earthquake, fire, etc.). In fact, this approach cannot detect such things as the death of a famous person.

Scalability or independence from space and time: The event detection system should not be dependent upon memory. This system should not slow down over time and should process each data upon arrival. This target states the consistency of the space required over time. Obviously, the system grows over time, which should not increase the required memory. Also, in solving a problem like event detection in Twitter, the growth rate is not fixed. In an approach like EDCoW [5] which works with signal processing, elimination of several tweets has been considered to reduce the space, while it cannot be a guarantee for scalability.

Real-time processing: This objective means the detection of the event as soon as it is observed. The real-time concept has been observed in a small number of approaches. Almost all of them conducted their assessment by collecting a dataset, but there is no discussion about the processing time in a real environment. In TwitterMonitor [19],the system works online, but even in this case, only a small number of tweets are tested due to lack of access to all tweets.

No supervision: The final event detection system must be able to operate without user monitoring, verification, and feedback. There should be no need for a training phase in the system since training is reserved for some tweets in the dataset whose behavior will change over time.

No use of a dictionary or additional data: Approaches such as [20] employed a dictionary as well to improve the accuracy of their system. The use of such things can result in a loss of generality of the approach because it is not possible to find and use a comprehensive dictionary for all the languages. Even if one finds and uses all the dictionaries, the system performance is reduced due to their high volume. Besides, the common languages in the world are generally dynamic and produce new words, so the use of a dictionary can be considered a weakness of the method.

2.3 Quality of Services in Event Detection

An effective event detection system is the one with a high recall to be able to retrieve each related event, as well as high precision to reject all the unrelated events [21].

In an event detection system, precision represents the usefulness of the output list (diagnostic events). The closer the events in the system output to real answers, the better this criterion. The precision is introduced as follows:

$$\text{Precision} = \frac{CDE}{TDE}. \tag{1}$$

In this equation, CDE (Correctly Detected Events) represents the number of correctly detected events, and TDE (Total Detected Events) is the total number of detected events. The recall represents the completeness of the output list. The recall is equal to:

$$\text{Recall} = \frac{CDE}{TE}. \tag{2}$$

In this equation, TE (Total Events) is equal to the total number of events that occurred during the period in question. Recall shows how well the system works in finding the events.

3 RELATED WORKS

Atefeh and Khreich generally classified event detection approaches in Twitter into document-pivot and feature-pivot techniques depending on whether they rely on a document or temporal features [22]. However, another classification has been recently proposed by Hasan et al. on Twitter-centric event detection systems [11].

Hasan et al. classified the event detection approach based on their common features into term-interestingness, topic-modeling, and incremental-clustering. The term-interestingness methods rely on tracking the terms from the Twitter data stream likely to be related to an event. The topic-modeling approaches associate each tweet with a probability distribution over the different latent topics to discover the hidden semantic structures from a stream of tweets to detect the related events. The incremental-clustering methods employ an incremental clustering strategy in order to avoid having a fixed number of clusters due to the high-volume, real-time Twitter data where a wide variety of topics are discussed.

Considering the above categories and taking into account the proposed approach in this article, we classify the research associated with Twitter event detection into the following two categories: Burst/Hot Keywords Frequency and Statistical Analysis.

A majority of event detection techniques use the frequency of keywords in users' posts to detect the related events. Some other approaches like Benhardus employ TF-IDF (Term Frequency–Inverse Document Frequency) factor, again related to the frequency. The extraction of Burst Keywords using Timing Windows can be used for event detection in many approaches, such as [19, 23, 24, 25]. The timing Window is a time slice in which the words are collected, and their sudden increase is studied. Also, looking for special keywords ("Hot Keywords") is another method to detect events. If the tweet or group of tweets include these hot keywords, then the associated event will be discovered (Section 3.1).

Some other approaches employ various statistical techniques such as LDA or Bayesian (Section 3.2). In these approaches, each tweet is associated with a probability distribution over various latent topics to discover the hidden semantic structures from a collection of tweets, such as [33, 34, 35].

3.1 Burst/Hot Keywords Frequency

As defined in articles, an event is expressed as a group of keywords, and the event detetcion system goal is to find this group that appears simultaneously in a stream of data.

A method for event detection using text mining techniques has been presented by Benhardus in [20]. This approach takes advantage of the frequency with TF-IDF factor as well as Entropy test. In this approach, the tweets are grouped in packages. Depending on when the tweets are sent, groups are normalized., which means that each document is linked to a fixed time interval.

Massive Online Analysis (MOA) TweetReader detects a trend in three steps [26]. In the first step, upon the arrival of each tweet from Twitter API, pre-processing is done, and then the feature vector is formed using TF-IDF parameter. In the second step, the tweets are tagged using a trained component, and finally, if a change is observed, the trend is discovered.

The SABESS (Social Awareness Based Emergency Situation Solver) approach employs the frequency method with some modifications [4]. Upon the arrival of each tweet, SABESS determines whether it is an urgent event or not, what category it is (flood, earthquake, fire, etc.), and where is the location of the event.

Recently, Frequent Pattern Mining (FPM) has been employed for event detection in Twitter. The FPM helps to find patterns of words that frequently occur in the Twitter data stream. The method called SFPM (Soft FPM) is also applied for this purpose by mitigating the requirement that all items must be frequent in the pattern [27]. Gaglio et al. extended the SFPM method to deal with dynamic environments of Twitter by splitting the streams into dynamic windows whose size depends both on the volume of tweets and time [28]. Huang et al. proposed an event detection based on the High Utility Pattern Clustering (HUPC) framework by clustering all patterns generated by the FP-Growth algorithm [29].

In the approach presented by Choi and Park, emerging topics on Twitter have been detected based on High Utility Pattern Mining (HUPM). The goal of HUPM is to find itemsets that have high frequency and high utility at the same time [30].

In [31], six different methods for topic detection have been evaluated (i.e., LDA, Doc-p, Gfeat-p, FPM, SFPM, and Bngram) on three datasets. Finally, a comparison of these methods concluded that the combination of DF-IDF and nGram in Bngram method showed the best results.

TwitterMonitor attempts to find the trend of users using the frequency method [19]. It first detects the Burst Keywords (keywords that suddenly appear in an unusually large number of tweets), and these words are then placed in their related groups. In other words, a trend is detected as a set of Burst Keywords that have frequently appeared together in tweets. Once a trend is detected, TwitterMonitor extracts additional information from the tweets related to the detected trend.

Guzman et al. have introduced an approach in [23] to detect sudden keywords. In this approach, suddenly rising words are detected using a five-stage algorithm. Each stage is written with a standalone module. Three modules are used for preprocessing, and the remaining two modules detect sudden words.

In EDCoW method, an event is indicated by keywords that are suddenly increased [5]. This approach attempts to detect new and important events using a signal processing technique. In EDCoW approach, a signal is generated only when a word shows a sudden behavior. The signal is then quickly processed without the need for considerable memory by Wavelet analysis. In fact, after receiving a signal, trivial words are discarded, and only the signal and its affiliates are considered. Then, the cross-correlation between signals is calculated, followed by event detection through signal clustering via graph partitioning.

TopicSketch is a framework for real-time detection of bursty topics on Twitter [32]. This approach utilizes two main techniques; a sketch-based topic model and a hashing-based dimension reduction technique. TopicSketch is not suitable for topic detection in a stream of documents with multiple topics.

Burnap et al. attempted to categorize public posts on Twitter according to their tension [3]. In this paper, tension is defined as follows: "any event that seriously disrupts a normal relationship between individuals or groups, which also spreads to groups or individuals not involved in the relationship". In their paper, first, the incoming tweet is tagged, and then the Burst keyword (if present) is detected in tension level. Their method determines the level of tension using simple rules. For example, if the tweet includes one or more words from vulgar and profane words, the tension level is detected as high.

Zhang et al. proposed a Pattern-based Topic Detection and Analysis System (PTDAS) on Weibo, a Chinese Twitter-like platform [33]. For this purpose, they have developed three different modules: Topic detection, Evolutionary analysis, and Sentimental analysis. A key component of their method is to employ an FP-growth-like algorithm to mine cosine interesting patterns from a set of tweets.

3.2 Statistical Analysis

Given a set of keywords, various statistical models such as pLSI (probabilistic Latent Semantic Indexing) [34] and LDA (Latent Dirichlet Allocation) [35] can be used for topic detection in Twitter data streams. However, pLSI was based on the likelihood principle, and it can not assign probabilities to new documents. This was resolved by LDA, which models each document as a mixture of topics and topics as a mixture of words. Indeed, LDA is a Bayesian network that generates a document using a mixture of topics and words.

In [13], the authors used the Online LDA approach, which has been developed for modeling several latent variables (titles) in a series of texts, including words. In this approach, new events make sense with new words (for example, names of people, parties, etc.), so that the collection of words is updated each time a document arrives. The words with a lower threshold are removed from the end.

Ahuja et al. have proposed Spatio-Temporal Event Detection (STED), a probabilistic model that detects events using information from news and Twitter [36]. For this purpose, they employed timestamps and geolocation information from tweets to estimate the temporal and regional distributions of events.

Huang et al. applied LDA to identify potential topics in a Twitter data stream [37]. They first employed ST-DBSCAN (an unsupervised data clustering algorithm) to cluster the tweets of every day. Moreover, spatial, temporal, and textual patterns for every cluster have been generated. Then, they applied LDA to identify potential topics in the cluster and analyze the structure of every tweet.

Gupta et al. collected the tweets and then, for unprocessed tweets run the LDA streaming and retrieved the tweet based on the domain to which it belongs [38]. If the tweet belongs to a certain event category, then it extracts tweets using domain-based classification. Moreover, the scoring function is used to correctly identify whether the tweet is belonging to that domain or not.

Another statistical model, namely Bursty Biterm Topic Model (BBTM), has been proposed by Yan et al. to solve the data sparsity problem in topic modeling over the short texts [39]. Their work is devoted to Biterm Topic Model (BTM), which models biterms (i.e., word pairs) rather than words for effective topic modeling in short texts.

Mehrotra et al. proposed an approach for aggregating tweets in order to improve the quality of LDA-based topic modeling in microblogs [40]. They achieved this through various pooling schemes that aggregate tweets in a data preprocessing step for LDA. Their pooling schemes included Author-wise, Burst-score-wise, Temporal, and Hashtag-based Pooling.

4 THE PROPOSED METHOD: MTCW

As discussed in the previous section, the use of Timing Windows is a convenient method for event detection. The Timing Window is a slice of time that could be a quarter of an hour, an hour, or even a day. If time slices are represented with k_t

then k_0 is the first slice of time. In Table 1, different time slices from ten minutes to one day have been shown.

Time slices	10 minutes	100 minutes	One day
k_0	From 1:00 AM to 1:10 AM	From 1:00 AM to 2:40 AM	From 2016-02-01 to 2016-02-02
k_1	From 1:10 AM to 1:20 AM	From 2:40 AM to 4:00 AM	From 2016-02-02 to 2016-02-03
k_2	From 1:20 AM to 1:30 AM	From 4:00 AM to 5:40 AM	From 2016-02-03 to 2016-02-04

Table 1. Examples of time slices

L is a Timing Window that keeps the words in a fixed number of time slices. When a new word arrives, the earlier word is removed from the beginning of the Timing Window. Therefore, the size of L will always be constant. Upon the arrival of a new word, the model is rebuilt. Table 2 presents an example of a Timing Window in size of 5, indicating that L can only contain five words; if a new word arrives, the earlier word is removed.

L in k_t		Content of L in size of 5 words				
L in k_0	Word	Iran	Rouhani	ART	100$	Planes
	Frequency	735	352	439	259	199
L in k_1	Word	Unfrozen	Claims	Iran	Rouhani	ART
	Frequency	689	598	497	356	241
L in k_2	Word	Unfrozen	Iran	Claims	Rouhani	ART
	Frequency	741	656	568	522	345
L in k_3	Word	Planes	Iran	Rouhani	Unfrozen	Claims
	Frequency	659	612	563	456	450

Table 2. Examples of timing window

The words in tweets of each time slice together with their frequency make up the content of a Timing Window. When a short slice of time is selected, a sharp increase in the number of certain words in this time slice indicates a sudden event.

We define several Timing Windows and link them to each other like chains in a way that the input of the larger window will be the output of the previous smaller window. This new concept has been introduced as Multi Timing Chained Windows (Figure 1).

A stop word is defined as a word that contains no meaning or relevance by itself [20]. In other approaches, a fixed list of Stop Words is usually used. Our approach, namely MTCW, is capable of simultaneously detecting sudden events and user events as well as making a dynamic list of Stop Words in the language of a query. The idea of dynamic generation of Stop Words helps the algorithm to be compatible with the environment under any conditions. More precisely, if a new word is spread among users, new words are added to the collection of Stop Words over time.

The developed approach for event detection comprises six modules (Figure 2). Each module has its specific task, and instead of saving its output, it directly delivers

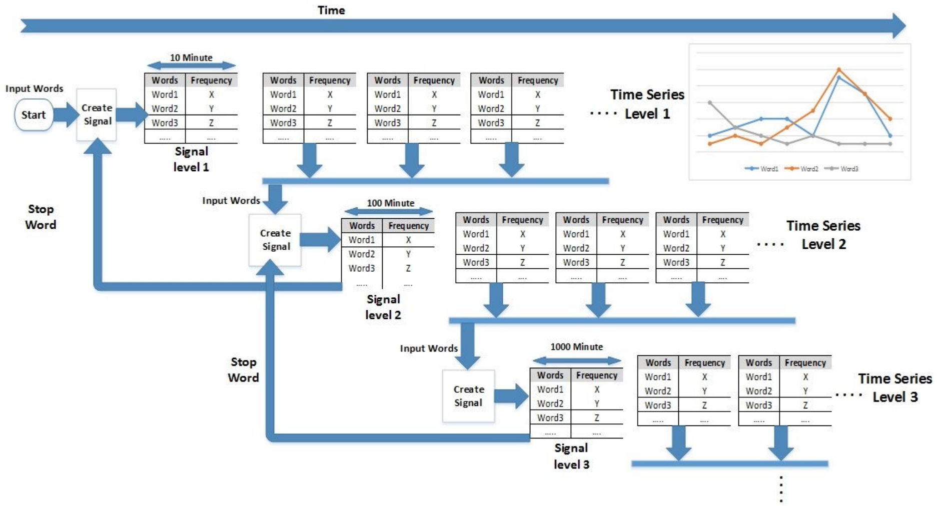


Figure 1. Different signal levels in MTCW approach

it to the next module to increase the speed and reduce the use of memory. In the following, each module is briefly explained.

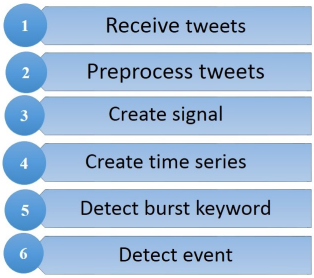


Figure 2. The layered architecture of proposed trend detection

4.1 The First Module: Receive Tweets

This module receives tweets from the Twitter Streaming API. The Twitter Streaming API continuously delivers the details of tweets requested by users in JSON format. Based on the track and location filters, this API only delivers a portion of tweets to the user.

Although MTCW, in the first step, collects tweets using Twitter Streaming API, the methodology of the proposed approach is not user-centric. In fact, user's tweets affect the functionality and the correctness of MTCW (same as all social-aware event detection systems), but the user/operator could not impose any preferences directly on the system's performances.

4.2 The Second Module: Preprocess Tweets

This module processes the text of a tweet in a straightforward way. In fact, every effort is made to spend the minimum possible time for initial processing. This module receives the text of a tweet and finally produces a sorted array of tweets along with the frequency. This module performs six operations on the text of a tweet, as shown in Pseudocode 1.

```
TweetPreprocess(string s){
    String r;
    Array result[][];
    r = ConvertLowCase(s);
    r = ConvertSpace(r);
    r = SplitBasedonSpace(r);
    r = RemoveWordsLessthan3char(r);
    for (i=0; i< length(r); i++)
        if (r[i] isInArray result)
            result[r[i]]++;
        else
            add r[i], 1 to result;
    AsortArray(result);
    return result;
}
```

Pseudocode 1: Preprocess tweets module

4.3 The Third Module: Create Signal

The assorted array of words from tweets and the frequency of each word is called signal. The signal is assorted in a descending order based on the frequency. In this study, each signal is constructed in four levels. In fact, four different Timing Windows are considered to construct a signal.

The first level signal: It should be noted that the first signal is received directly from the words of the first Timing Window tweets, while in the next levels, words from a previous level are used as the input. In this level, tweets are received in a ten-minute interval, and their text is delivered to the second module for processing. The arrays received from the second module are combined with each other to produce the signal.

The process is as follows: words in the fourth level of Timing Windows (Dynamic Stop Words) are deleted from arrays. Then, the words in the next level time series (second level) are also deleted from the array. In fact, the words that have reached the next level are those with a high frequency in the recent past, which cannot thus represent an urgent event. Frequencies of similar words are added together. Afterwards, the remaining words of the signal are arranged in descending order based on the sum of frequencies. Finally, the signal is sliced according to first level limitation parameter of the signal word. In MTCW, this parameter is set to 100 for the first level signal, i.e., 100 frequently repeated words are maintained. In this way, space limitation is observed, and there is no need for further space with an increasing number of incoming tweets.

The pseudocode of the above process has been listed in Pseudocode 2.

```
CreateSignals(inputArray, lastLevelWords, nextLevelWords, limitationSignalLevel){

        inputArray = LastLevelRemoveWords (inputArray, lastLevelWords);
        inputArray = nextLevelRemoveWords (inputArray, nextLevelWords);
        outArray = AsortArray (outArray);
        signal = CutArray (outArray , limitationSignalLevel);

        return result;
}
```

Pseudocode 2: Create signal module

Making the second, third, and fourth level signal: After few iterations of the first level signal, the second level signal is constructed. After generating ten signals in the first level, a new signal for the second level is constructed (every 100 minutes). This process is repeated to create next level signals. Indeed, after ten iterations of the second level signal, the third level signal is constructed (every 1 000 minutes), and after five iterations of third level signal, the fourth level signal is constructed.

The procedure for signal generation is as follows:

- Considering the current signal level, five to ten signals of the previous level are combined to create the signal of the next level.
- The signal integration is performed by summing up the total number of similar words (frequency).
- For the second level signals, the words that exist in the next time series (third) is deleted. The same happens for the third level signal.
- The result is assorted in a descending order based on the frequency.
- Finally, the assorted result is sliced based on the words limitations.

4.4 The Fourth Module: Create Time Series

A number of signals generated at successive time intervals is called time series. There is a distinct time series for each level. The series is updated upon the creation of each signal. The update is done as follows:

If a word is absent in a time series, it is added to the time series, and iteration is set to zero in previous signals for that word. If the word exists in the time series, a new iteration is added as the last word signal, and the oldest signal of that word is deleted. If both previous conditions are false, then a zero repeat is added to the word, and the earliest signal of that word is deleted.

The pseudocode for creating time series is listed in Pseudocode 3.

```
CreateTimeSeries(signals){
    array timeSeries[][];
    foreach (signals as signal){
        CreateNewSignalTimeSeries (timeSeries);
        foreeach (signal as word => frequency)
            if (word is not in timeSeries){
                AddToTimeSeries(timeSeries, word, frequency);
                AddZeroToLastSignalTimeSeries (timeSeries, word);
            } else{
                AddNewFrequencyWordToOld (timeSeries, word , newFrequency);
            }
        AddZeroOtherWordsNewSignal(timeSeries);
    }
}
```

Pseudocode 3: Create time series module

No time series is constructed for the fourth level, but its signal is updated. The reason for this is that Stop Words are kept at this level and that the highest iterations should always remain at this level. Therefore, upon the arrival of a new signal at the fourth level, the number of iterations is compared with the previous signal, and if there is a higher value in the new signal, the previous signal is updated. Finally, based on the limitation of signal words, only the most frequently repeated ones remain.

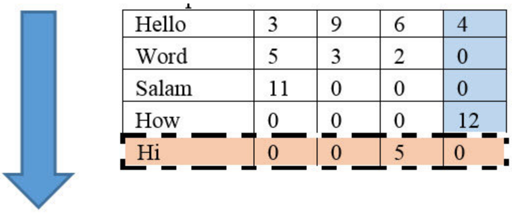
In other words, in the previous levels, the words are horizontally and vertically removed from Timing Windows, but in the fourth level, the words are removed vertically.

Vertical and horizontal update: The following time series in Table 3 includes four words and four-signal limitations. If the new signal arrives with a "How" word and iteration of 12, the word "Hi" is excluded due to the limitation of words. In this case, we say the word was out vertically (Figure 3).

Now, suppose the arrival of a new signal, including the word "Hello" with five iterations. In this case, all iterations of the word "Salam" are set to zero and are deleted from the set of time series words. Now, we say the word was out horizontally (Figure 4).

Hello	0	3	9	6
Word	2	5	3	2
Salam	0	11	0	0
Hi	1	0	0	5

Table 3. Time series in the last level



Hello	3	9	6	4
Word	5	3	2	0
Salam	11	0	0	0
How	0	0	0	12
Hi	0	0	5	0

Figure 3. Vertical update of the last level time series

4.5 The Fifth Module: Detect Burst Keywords

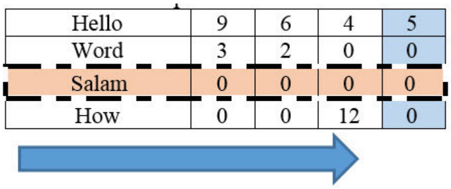
After updating the time series, the fifth module investigates a sudden increase in the iteration of words. The output of this module is the candidate of Burst Keywords.

The following criterion is used to find candidate Burst keywords:

$$\left|\frac{F_w - M_w}{M_w}\right| > BP. \tag{3}$$

In this formula,F_w represents the frequency of the word W in the new signal. M_w is the average frequency of the word W in previous signals. In addition, BP (burst parameter) is a measure of Burst Keyword that can be different for each level. In our experiments, the BP value has been experimentally considered 0.3 for the first level and 0.7 for the second and third levels, respectively.

For example, if the frequency of the word "planes" in the first level of the new signal is 1 200 ($F_w = 1\,200$) and the average frequency value of this word in previous signals is 1 000 ($M_w = 1\,000$), then this word is a candidate as a Burst Keyword in the first level of the signal ($BP = 0.3 > 0.2$). The candidate Burst Keywords are



Hello	9	6	4	5
Word	3	2	0	0
Salam	0	0	0	0
How	0	0	12	0

Figure 4. Horizontal update of the last level time series

returned together with their entire signal to calculate the similarity of their keywords in the next module.

4.6 The Sixth Module: Detect Event

We detect events by grouping a set of candidate Burst Keywords with similar burst patterns. This set includes a few candidate Burst Keywords with a sudden increase in their iteration with a similar pattern of burst in recent signals.

The similarity in the time series of candidate Burst Keywords is checked for each signal. Finding the level of similarity in the time series of two candidate Burst Keywords can indicate the similarity of their iteration pattern. If the two candidate Burst Keywords have the same iteration pattern, it can be stated that they are at the same set.

The cross-correlation measure is used to examine the behavioral similarity of two candidate Burst Keywords. The cross-correlation is a criterion to detect the similarity of two time series in signal processing. In the discrete domain, the following equation is used to calculate the cross-correlation for two time series of x and y with the length of n [41]:

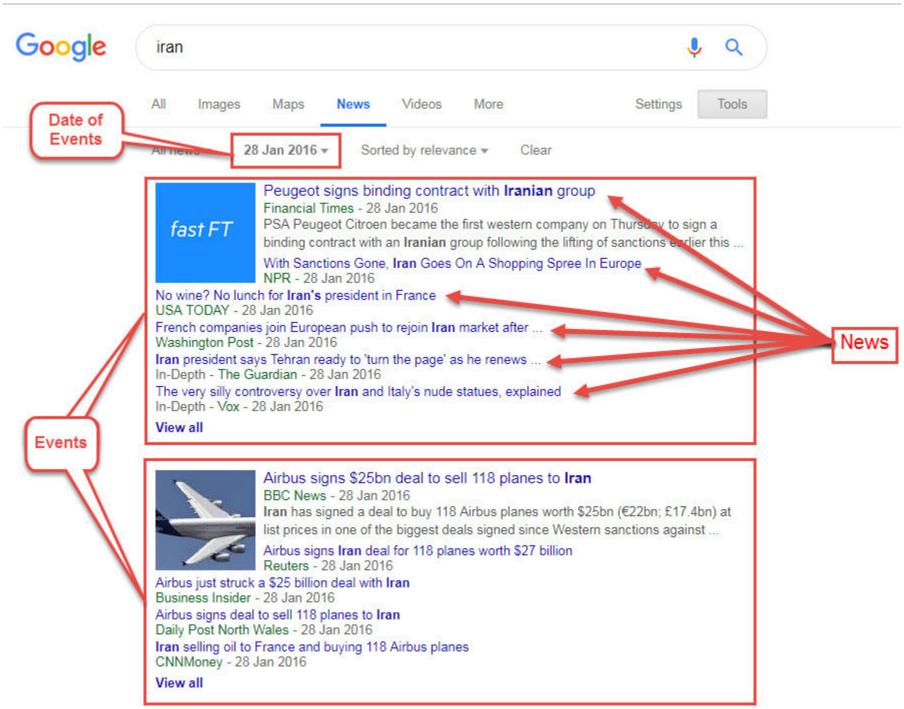
$$r = \frac{\sum_i^n [(x(i) - mx) \times (y(i-d) - my)]}{\sqrt{\sum_i^n (x(i) - mx)^2} \sqrt{\sum_i^n y(i-d) - my)^2}} \tag{4}$$

where x and y are actually two time series supposed to be measured in terms of similarity, with m_x and m_y presenting the mean of two time series. d is the lag parameter for which the value of 1 has been chosen in this algorithm.

5 EXPERIMENTAL RESULTS

A dataset has been collected using Twitter streaming API to analyze and evaluate this idea. The tweets were collected from 28. 1. 2016 to 27. 2. 2016 based on track = “iran”, “tehran”, “thr”. Over one and a half million tweets (1 524 493) were collected. Tweets from the same day were classified together, and every day was divided into ten-minute intervals. For each interval, the tweets were stored as a text file. The information collected from every tweet included ID, date, text, origin, User ID, username, and location of the user posting the tweet.

It should be mentioned that most proposed approaches in event detection use their proprietary datasets for evaluation purposes. To this end, Twitter streaming API has been employed to collect and create our proprietary dataset. In fact, to evaluate MTCW, we compared the News items related to Iran, which were detected by our proposed approach with the Google News search service. The search was done by examining Google News, searching for the word Iran, and setting the time interval from day to day. However, several methods benefit from publicly available datasets. In order to precisely evaluate MTCW and eliminate any possible inconsistency or bias in the dataset, MTCW has also been evaluated using a publicly available dataset, which has been employed by many articles. The results are reported in Section 6.2.



28 Jan 2016 ▾ Sorted by relevance ▾ Clear

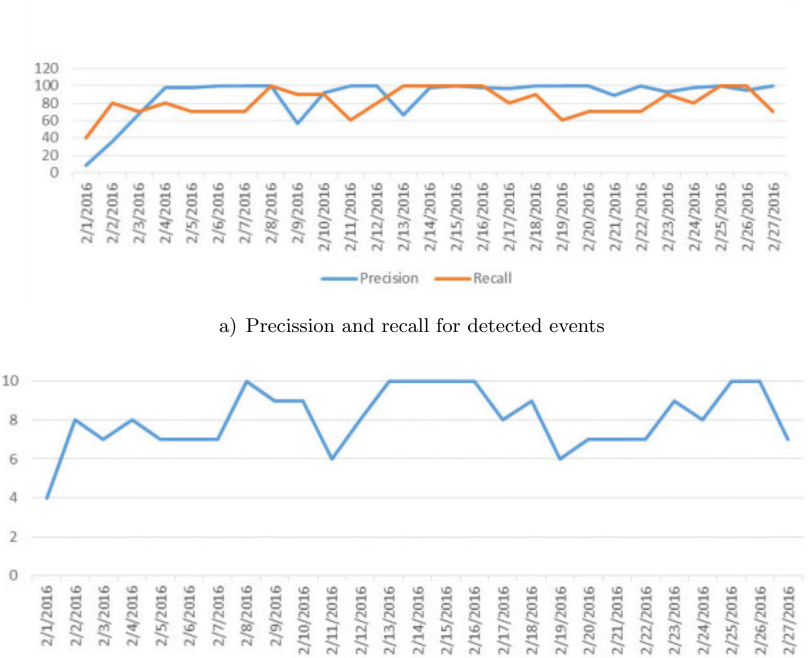
Peugeot signs binding contract with **Iranian** group
Financial Times - 28 Jan 2016
PSA Peugeot Citroen became the first western company on Thursday to sign a binding contract with an **Iranian** group following the lifting of sanctions earlier this ...
With Sanctions Gone, **Iran** Goes On A Shopping Spree In Europe
NPR - 28 Jan 2016
No wine? No lunch for **Iran's** president in France
USA TODAY - 28 Jan 2016
French companies join European push to rejoin **Iran** market after ...
Washington Post - 28 Jan 2016
Iran president says Tehran ready to 'turn the page' as he renews ...
In-Depth - The Guardian - 28 Jan 2016
The very silly controversy over **Iran** and Italy's nude statues, explained
In-Depth - Vox - 28 Jan 2016
View all

Airbus signs $25bn deal to sell 118 planes to **Iran**
BBC News - 28 Jan 2016
Iran has signed a deal to buy 118 Airbus planes worth $25bn (€22bn; £17.4bn) at list prices in one of the biggest deals signed since Western sanctions against ...
Airbus signs **Iran** deal for 118 planes worth $27 billion
Reuters - 28 Jan 2016
Airbus just struck a $25 billion deal with **Iran**
Business Insider - 28 Jan 2016
Airbus signs deal to sell 118 planes to **Iran**
Daily Post North Wales - 28 Jan 2016
Iran selling oil to France and buying 118 Airbus planes
CNNMoney - 28 Jan 2016
View all

Figure 5. Google News service

As mentioned above, for the first phase of evaluation, Google New search service returns ten events related to Iran on a respective day. An event may include 1–7 news items from different News agencies. The collection of a number of news from an event contributes to a better analysis of the results. We may have detected an event but with different words and syntax. Different headlines from various news agencies concerning a single event are an idea for a better comparison of results. We combine these headlines, and if the detected event is close to this combination, then we declare the detection of that event.

The results of the evaluation related to the first level series and Google News headlines are shown in Figure 5. As it can be seen in Figure 6 b, for some days, all ten events have been detected. Also, the results have been improved over time.



a) Precission and recall for detected events

b) Detected events day by day

Figure 6. Experimental results

6 EVALUATION AND COMPARISON

In Section 5, we provided the precision and recall of our event detection approach compared to the Google News service. For further evaluation of the proposed method, quantitative and qualitative comparisons have been performed with other state-of-the-art methods in event detection in Twitter, as discussed in the next subsections.

6.1 Qualitative Comparison

In the following Table, the approaches assessed in the Related Work section have been compared based on the five objectives described in Section 2.2.

MTCW approach has the Generality feature. Since the algorithm is not sensitive to a specific subject and no word or phrase has already been tagged, it recognizes an event based on frequency patterns of words (which can be related to each topic).

Our solution does not depend on time and space (Scalability feature). In the proposed algorithm, the signal length is constant at all levels, so the algorithm only needs a constant space for storing information and the relevant history, and this space will not increase over time. The processing done in the algorithm does not change over time due to the constancy of the input, so the complexity of the time is constant.

The proposed solution requires limited preprocessing, and little time is needed to process the tweets and generate the result (Real-time processing feature). In fact, the algorithm produces the result of an event within the specified time interval without any delay at each signal level.

The use of the idea of dynamic Stop Words forms an unsupervised system (No Supervision feature). In this system, based on user behavior, meaningless words are removed over time, and meaningful words are added to the system. MTCW does not use any dictionary or additional words and data, either.

No.	**Approach**	**Generality**	**Scalability**	**Real-time processing**	**No supervision**	**Non-use Additional Data**
1	Benhardus [20]	✓	✓	✗	✓	✗
2	TwitterMonitor [19]	✓	✓	✓	✓	✓
3	EDCoW [5]	✓	✓	✓	✓	✓
4	SasaPetrovic [42]	✗	✓	✓	✓	✓
5	Bifet [26]	✗	✗	✗	✓	✓
6	Pete Burnap [3]	✗	✓	✗	✗	✗
7	Takeshi Sakaki [43]	✗	✗	✓	✓	✗
8	Aielloi [31]	✓	✗	✓	✗	✗
9	Hyeok Jun Choi [30]	✓	✓	✓	✓	✓
10	MTCW	✓	✓	✓	✓	✓

Table 4. Qualitative comparison of different approaches

6.2 Quantitative Comparison

To evaluate MTCW, we utilized the same evaluation framework proposed by Aiello et al. [31]. They have extracted tweets about three major real-world events that occurred in 2012, which includes the FA Cup Final (FA), Super Tuesday (ST), and US Elections (US). Since they are not allowed to publicly distribute the original tweets, the distributions only contain the tweet IDs and the ground truth topics which have been organized in timeslots as explained in [1]. Indeed, Aiello et al. generated a ground truth for the dataset consisting of a manual review of published media reports about the event. This ground truth includes 13 topics for FA, 22 topics for ST, and 64 topics for US datasets [31].

The "FA Cup" dataset contains tweets posted during the final match of the Football Association Challenge Cup held on May 5, 2012. The ground truth for the FA Cup dataset comprises 13 topics, including kick off, goals, half-time, fouls, bookings, and the end of the match.

The "Super Tuesday" dataset consists of tweets posted during the US primary elections, which were held on the first Tuesday of March 2012 in ten US states. The ground truth comprises 22 topics, which represent the key moments of the elections and projections of the voting results in different states.

The "US Election" dataset contains tweets posted during the United States presidential election of 2012, which was held on November 6, 2012. The ground truth consists of 64 topics. The topics were related to the outcomes of the presidential election, derived from mainstream media.

Totally, we extracted about 200k tweets for FA, 500 k tweets for ST, and 1 100 k tweets for the US. It should be mentioned that some tweets were not downloaded as they are not available anymore. Sequiera and Lin [44] performed experiments on the long term effect of tweet removal from Tweets2013 corpus. They observed that the deletions would less likely have an impact on the ranking of systems. The details of the three datasets are given in Table 5.

Dataset	**Temporal Coverage**	**No. of Tweets**	**Total Topics**
FA Cup	6 HOURS	124 524	13
Super Tuesday	24 HOURS	540 241	22
US Election	36 HOURS	2 335 105	64

Table 5. Datasets details

We have compared the precision and recall of MTCW with some well-known methods based on the above datasets in Table 6. The baselines selected for evaluation include the state-of-the-art event detection systems and cover a wide range of techniques in this domain, such as BNgram [31], Frequent Pattern Mining (FPM) [31], Soft Frequent Pattern Mining (SFPM) [27], Graph-based feature-pivot (GFeat-p) [31], and a probabilistic topic model-based LDA [45]. Moreover, we have compared the performance of MTCW with a recently published method, HUPM [30]. As shown in Table 6, our approach proposes competitive or even better results with state-of-the-art event detection approaches.

The experimental results for the FA dataset show that both precision and recall are the highest for MTCW. Similarly, for the ST dataset, MTCW precision is the best one after FPM, and MTCW recall is very close to SFPM, which offers the best recall among others. The evaluation results from the US dataset indicates that MTCW precision is the highest, and its recall is very competitive among the compared methods.

Method	**FA Cup**			**Super Tuesday**			**US Elections**		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score
BNgram	0.2989	0.5778	0.394	0.6286	0.6471	0.6377	0.4050	0.5632	0.4712
FPM	0.7500	0.4286	05455	1.0000	0.4091	05807	0.0000	0.0000	0
SFPM	0.2336	0.6579	0.3448	0.4717	0.8929	0.6173	0.2412	0.6953	0.3582
GFeat-p	0.0000	0.0000	0	0.3750	0.6000	0.4615	0.3750	0.4839	0.4225
LDA	0.1637	0.6829	0.2641	0.0000	0.0000	0	0.1654	0.6286	0.2619
HUPM	0.3200	0.6000	0.4174	0.4860	0.7080	0.5764	0.3520	0.5650	0.4338
MTCW	0.8500	0.7250	0.7825	0.7750	0.8234	0.7985	0.4650	0.6125	0.5287

Table 6. Quantitative comparison of different approaches

7 CONCLUSIONS

In this paper, we proposed a new approach for event detection in Twitter using Multi Timing Chained Windows (MTCW). In our method, the time and space complexity are constant for any number of tweets, and the approach is also independent of user language. We examined MTCW on Iran-related tweets over a period of 27 days. More than 1.5 million tweets related to Iran were collected in this period. Using the Google News service, news about Iran were categorized within the period of tweets collection, which were used to evaluate the results. The results indicate the high precision of the proposed method. Moreover, common datasets such as FA Cup Final, Super Tuesday, and US Elections have been employed to compare MTCW with baselines and recent approaches.

As future works, it is suggested to collect the tweets based on their location instead of their subject to improve the effectiveness of the results, but we should be aware of the complexity of this dataset. In addition, the TF-IDF approach can be used instead of frequency to create signals. Also, by increasing the number of windows, a closer examination of the precision of this method will become possible.

REFERENCES

[1] Twitter by the Numbers. 2021, available at: `https://www.omnicoreagency.com/twitter-statistics`.

[2] Company – About – Twitter. 2021, available at: `https://about.twitter.com/company`.

[3] Burnap, P.—Rana, O. F.—Avis, N.—Williams, M.—Housley, W.—Edwards, A.—Morgan, J.—Sloan, L.: Detecting Tension in Online Communities with Computational Twitter Analysis. Technological Forecasting and Social Change, Vol. 95, 2015, pp. 96–108, doi: 10.1016/j.techfore.2013.04.013.

[4] Klein, B.—Castanedo, F.—Elejalde, I.—López-de-Ipiña, D.—Prada Nespral, A.: Emergency Event Detection in Twitter Streams Based on Natural Language Processing. In: Urzaiz, G., Ochoa, S. F., Bravo, J., Chen, L. L., Oliveira, J. (Eds.): Ubiquitous Computing and Ambient Intelligence. Context-Awareness and Context-Driven Interaction. Springer, Cham, Lecture Notes in Computer Science, Vol. 8276, 2013, pp. 239–246, doi: 10.1007/978-3-319-03176-7_31.

[5] Weng, J.—Yao, Y.—Leonardi, E.—Lee, F.: Event Detection in Twitter. Development, 2011, pp. 401–408.

[6] Gerber, M. S.: Predicting Crime Using Twitter and Kernel Density Estimation. Decision Support Systems, Vol. 61, 2014, pp. 115–125, doi: 10.1016/j.dss.2014.02.003.

[7] Dong, G.—Yang, W.—Zhu, F.—Wang, W.: Discovering Burst Patterns of Burst Topic in Twitter. Computers and Electrical Engineering, Vol. 58, 2017, pp. 551–559, doi: 10.1016/j.compeleceng.2016.06.012.

[8] Petrović, S.—Osborne, M.—McCreadie, R.—Macdonald, C.—Ounis, I.—Shrimpton, L.: Can Twitter Replace Newswire for Breaking News? Proceedings of

the Seventh International AAAI Conference on Weblogs and Social Media (ICWSM-13), Boston, MA, USA, 2013.

[9] Osborne, M.—Dredze, M.: Facebook, Twitter and Google Plus for Breaking News: Is There a Winner? Proceedings of 8^{th} International Conference on Weblogs and Social Media (ICWSM 2014), 2014, pp. 611–614.

[10] Allan, J.—Lavrenko, A. V.—Jin, H.: First Story Detection in TDT Is Hard. Proceedings of the Ninth International Conference on Information and Knowledge Management (CIKM 2000), 2000, pp. 374–381, doi: 10.1145/354756.354843.

[11] Hasan, M.—Orgun, M. A.—Schwitter, R.: A Survey on Real-Time Event Detection from the Twitter Data Stream. Journal of Information Science, Vol. 44, 2018, No. 4, pp. 443–463, doi: 10.1177/0165551517698564.

[12] Allan, J.: Topic Detection and Tracking: Event-Based Information Organization. Kluwer Academic Publishers, 2002, doi: 10.1007/978-1-4615-0933-2.

[13] Lau, J. H.—Collier, N.—Baldwin, T.: On-Line Trend Analysis with Topic Models: #twitter Trends Detection Topic Model Online. International Conference on Computational Linguistics (COLING 2012), Vol. 2, 2012, pp. 1519–1534.

[14] Rafea, A.—GabAllah, N. A.: Topic Detection Approaches in Identifying Topics and Events from Arabic Corpora. Procedia Computer Science, Vol. 142, 2018, pp. 270–277, doi: 10.1016/j.procs.2018.10.492.

[15] Madani, A.—Boussaid, O.—Zegour, D. E.: Real-Time Trending Topics Detection and Description from Twitter Content. Social Network Analysis and Mining, Vol. 5, 2015, Art. No. 59, doi: 10.1007/s13278-015-0298-5.

[16] Gaglio, S.—Lo Re, G.—Morana, M.: A Framework for Real-Time Twitter Data Analysis. Computer Communications, Vol. 73, 2016, Part B, pp. 236–242, doi: 10.1016/j.comcom.2015.09.021.

[17] Comito, C.—Forestiero, A.—Pizzuti, C.: Bursty Event Detection in Twitter Streams. ACM Transactions on Knowledge Discovery from Data, Vol. 13, 2019, No. 4, Art. No. 44, 28 pp., doi: 10.1145/3332185.

[18] 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 (ICWSM), 2011.

[19] Mathioudakis, M.—Koudas, N.: TwitterMonitor: Trend Detection over the Twitter Stream. Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data (SIGMOD '10), 2010, pp. 1155–1158, doi: 10.1145/1807167.1807306.

[20] Benhardus, J.—Kalita, J.: Streaming Trend Detection in Twitter. International Journal of Web Based Communities (IJWBC), Vol. 9, 2013, No. 1, pp. 122–139, doi: 10.1504/ijwbc.2013.051298.

[21] Fawcett, T.: An Introduction to ROC Analysis. Pattern Recognition Letters, Vol. 27, 2006, No. 8, pp. 861–874, doi: 10.1016/j.patrec.2005.10.010.

[22] Atefeh, F.—Khreich, W.: A Survey of Techniques for Event Detection in Twitter. Computational Intelligence, Vol. 31, 2015, No. 1, pp. 132–164, doi: 10.1111/coin.12017.

[23] Guzman, J.—Poblete, B.: On-Line Relevant Anomaly Detection in the Twitter Stream: An Efficient Bursty Keyword Detection Model. Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description (ODD '13), 2013, pp. 31–39, doi: 10.1145/2500853.2500860.

[24] Zhang, C.—Lei, D.—Yuan, Q.—Zhuang, H.—Kaplan, L.—Wang, S.—Han, J.: GeoBurst+: Effective and Real-Time Local Event Detection in Geo-Tagged Tweet Streams. ACM Transactions on Intelligent Systems and Technology, Vol. 9, 2018, No. 3, Art. No. 34, 24 pp., doi: 10.1145/3066166.

[25] Zhang, C.—Zhou, G.—Yuan, Q.—Zhuang, H.—Zheng, Y.—Kaplan, L.—Wang, S.—Han, J.: GeoBurst: Real-Time Local Event Detection in Geo-Tagged Tweet Streams. Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16), 2016, pp. 513–522, doi: 10.1145/2911451.2911519.

[26] Bifet, A.—Holmes, G.—Pfahringer, B.: MOA-TweetReader: Real-Time Analysis in Twitter Streaming Data. In: Elomaa, T., Hollmén, J., Mannila, H. (Eds.): Discovery Science (DS 2011). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 6926, 2011, pp. 46–60, doi: 10.1007/978-3-642-24477-3_7.

[27] Petkos, G.—Papadopoulos, S.—Aiello, L.—Skraba, R.—Kompatsiaris, Y.: A Soft Frequent Pattern Mining Approach for Textual Topic Detection. Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS '14), 2014, Art. No. 25, 10 pp., doi: 10.1145/2611040.2611068.

[28] Gaglio, S.—Lo Re, G.—Morana, M.: Real-Time Detection of Twitter Social Events from the User's Perspective. 2015 IEEE International Conference on Communications (ICC), 2015, pp. 1–6, doi: 10.1109/icc.2015.7248487.

[29] Huang, J.—Peng, M.—Wang, H.: Topic Detection from Large Scale of Microblog Stream with High Utility Pattern Clustering. Proceedings of the 8th Workshop on Ph.D. Workshop in Information and Knowledge Management, 2015, pp. 3–10, doi: 10.1145/2809890.2809894.

[30] Choi, H.-J.—Park, C. H.: Emerging Topic Detection in Twitter Stream Based on High Utility Pattern Mining. Expert Systems with Applications, Vol. 115, 2019, pp. 27–36, doi: 10.1016/j.eswa.2018.07.051.

[31] Aiello, L. M.—Petkos, G.—Martin, C.—Corney, D.—Papadopoulos, S.—Skraba, R.—Göker, A.—Kompatsiaris, I.—Jaimes, A.: Sensing Trending Topics in Twitter. IEEE Transactions on Multimedia, Vol. 15, 2013, No. 6, pp. 1268–1282, doi: 10.1109/tmm.2013.2265080.

[32] Xie, W.—Zhu, F.—Jiang, J.—Lim, E.-P.—Wang, K.: TopicSketch: Real-Time Bursty Topic Detection from Twitter. IEEE Transactions on Knowledge and Data Engineering, Vol. 28, 2016, No. 8, pp. 2216–2229, doi: 10.1109/tkde.2016.2556661.

[33] Zhang, L.—Wu, Z.—Bu, Z.—Jiang, Y.—Cao, J.: A Pattern-Based Topic Detection and Analysis System on Chinese Tweets. Journal of Computational Science, Vol. 28, 2018, pp. 369–381, doi: 10.1016/j.jocs.2017.08.016.

[34] Hofmann, T.: Probabilistic Latent Semantic Indexing. Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99), 1999, pp. 50–57, doi: 10.1145/312624.312649.

[35] Blei, D. M.—Ng, A. Y.—Jordan, M. I.: Latent Dirichlet Allocation. Journal of Machine Learning Research, Vol. 3, 2003, pp. 993–1022.

[36] Ahuja, A.—Baghudana, A.—Lu, W.—Fox, E. A.—Reddy, C. K.: Spatio-Temporal Event Detection from Multiple Data Sources. In: Yang, Q., Zhou, Z. H., Gong, Z., Zhang, M. L., Huang, S. J. (Eds.): Advances in Knowledge Discovery and Data Mining (PAKDD 2019). Springer, Cham, Lecture Notes in Computer Science, Vol. 11439, 2019, pp. 293–305, doi: 10.1007/978-3-030-16148-4_23.

[37] Huang, Y.—Li, Y.—Shan, J.—Huang, Y.—Li, Y.—Shan, J.: Spatial-Temporal Event Detection from Geo-Tagged Tweets. ISPRS International Journal of Geo-Information, Vol. 7, 2018, No. 4, Art. No. 150, 21 pp., doi: 10.3390/ijgi7040150.

[38] Gupta, M.—Gupta, P.: Research and Implementation of Event Extraction from Twitter Using LDA and Scoring Function. International Journal of Information Technology, Vol. 11, 2019, No. 2, pp. 365–371, doi: 10.1007/s41870-018-0206-0.

[39] Yan, X.—Guo, J.—Lan, Y.—Xu, J.—Cheng, X.: A Probabilistic Model for Bursty Topic Discovery in Microblogs. Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI '15), 2015, pp. 353–359.

[40] Mehrotra, R.—Sanner, S.—Buntine, W.—Xie, L.: Improving LDA Topic Models for Microblogs via Tweet Pooling and Automatic Labeling. Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '13), 2013, pp. 889–892, doi: 10.1145/2484028.2484166.

[41] Montgomery, D. C.—Jennings, C. L.—Kulahci, M.: Introduction to Time Series Analysis and Forecasting. Second Edition. Wiley, 2015.

[42] Petrović, S.: Real-Time Event Detection in Massive Streams. Ph.D. Thesis, University of Edinburgh, 2012.

[43] Sakaki, T.—Okazaki, M.—Matsuo, Y.: Earthquake Shakes Twitter Users: Real-Time Event Detection by Social Sensors. Proceedings of the 19th International Conference World Wide Web (WWW '10), 2010, pp. 851–860, doi: 10.1145/1772690.1772777.

[44] Sequiera, R.—Lin, J.: Finally, a Downloadable Test Collection of Tweets. Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17), 2017, pp. 1225–1228, doi: 10.1145/3077136.3080667.

[45] Teh, Y. W.—Newman, D.—Welling, M.: A Collapsed Variational Bayesian Inference Algorithm for Latent Dirichlet Allocation. In: Schölkopf, B., Platt, J., Hoffman, T. (Eds.): Advances in Neural Information Processing Systems 19 (NIPS 2006), 2006, pp. 1353–1360.

Mohammad Mahdi Mojiri received his B.Sc. degree in software engineering from the University of Kashan, Iran, in 2008 and his M.Sc. degree in software engineering from the Central Tehran Branch, Islamic Azad University, Tehran in 2017. His research interests include data mining, social network analysis, and stream processing.

Reza Ravanmehr graduated in computer engineering from the Shahid Beheshti University, Tehran, in 1996. After that, he gained M.Sc. and Ph.D., both in computer engineering from the Islamic Azad University, Science and Research Branch, Tehran, in 1999 and 2004, respectively. His main research interests are distributed/parallel systems, large-scale data management systems, and social network analysis. He is a faculty member of the Computer Engineering Department at the Central Tehran Branch, Islamic Azad University, since 2001.