The Journal of Supercomputing (2021) 77:11228–11256
https://doi.org/10.1007/s11227-021-03717-4

Burst: real-time events burst detection in social text stream

Tajinder Singh[1] · Madhu Kumari[2]

Accepted: 25 February 2021 / Published online: 22 March 2021

Abstract

Gigantic growth of social media and unbeatable trend of progress in the direction of the web seeking user's interests have generated a storm of social text streams. Seeking information to know the phenomenon of various events in the early stages is quite interesting. Various kinds of social media live streams attract users to participate in real-time events to become a part of an immense crowd. However, the vast amount of text is present on social media, the unnecessary information bogs a social text stream filtering to extract the appropriate topics and events effectively. Therefore, detecting, classifying, and identifying burst events is quite challenging due to the sparse and noisy text of Twitter. The researchers' significant open challenges are the effective cleaning and profound representation of the text stream data. This research article's main contribution is to provide a detailed study and explore bursty event detection in the social text stream. Thus, this work's main motive is to present a concise approach that classifies and detects the event keywords and maintains the record of the event based on related features. These features permit the approach to successfully determine the booming pattern of events scrupulously at different time span. Experiments are conducted and compared with the state-of-the-art methods, which reveals that the proposed approach is proficient to detect valuable patterns of interest and also achieve better scoresto extract burst events on social media posted by various users.

Keywords Event · Text stream · Text normalization · Social media · Online clustering · Decision-making

✉ Tajinder Singh
nith2k14@gmail.com

Madhu Kumari
madhu.jaglan@gmail.com

1 Sant Longowal Institute of Engineering & Technology, Sangrur, Punjab, India

2 Indian Institute of Management, Amritsar, India

1 Introduction

With the increase in social media, social networks' gigantic growth is being rapidly used by both journalists and customers alike [1]. Social media has changed the way of thinking and provides a different way to understand the globe today [2]. As social media is dynamic and information is posted with precise timestamps. With this, everything posted over social media changes over time [2, 3]. Each minute numerous multimedia (text, image, videos) is shared abundantly. Such sharing pattern of streaming information contains multiple events around the globe [4]. Such events are the primary source of information to gain and know about the most recent happenings, either huge or diminutive, around the world. Event "E" is defined as a happening [5] which have occurred or may occur in the next time span [6]. Multiple feasible ways are available for classifying or correlating a social media testimony to a particular event [3, 7–10]. In describing an event, consumers use #tag (hashtag) or @ symbol [11], which indicates that the content of particular events may be coupled with several events directly or indirectly [12]. A directed acyclic graph describes the relationship between two events. Suppose an event E occurs at a time $t(e_t)$,and a user u_n is tagged by a random user in an event stream using tags (# or @, etc.). A relationship may be directly or indirectly related to a particular user u_n that influences other users' existence and behavior, which is described in Eq. 1.

$$E = \{e_t \propto u_n\} \forall U \text{ where } u_n \propto e_t \; e_t \subseteq E \text{ and } u_n \subseteq U \tag{1}$$

In a stream of text, events are the real-world's happening, including a timec_t. Numerous userspost on features related to the event E. This scenario will be in a continuous fashion for a period known as event start time E_{s_t} and event close E_{c_t} time. In text stream, it is assumed that the stream documents are considerably associated with an event's features extracted within the closed time interval of time $[t_1, t_2]$, i.e., T. In other words, we can say that, before extraction text stream, event-related features can be posted on social media. Two types of events occur in social media, which are known as planned and unplanned events. It is observed that planned and trendy/bursy events are not commonly exclusive events; therefore, an event simultaneously either be trendy or non-trendy and can be either planned or unplanned, which is directly related to the event's frequency. $E_{burst} \propto \xi$ where ξ represents the frequency of a particular event. Planned events such as ICC World Twenty20 are an example of a bursty event whereas, Indian Idol Premier is also planned event but is not bursty.

Similarly, unplanned events such as Hurricane 2017, USA was bursty, and the minor road accident is non-bursty in the different span. The bursty behavior is directly associated with the diffusion rate of information over a social media or network. It has concerned and motivated numerous researchers to participate in this emerging area to extract and analyze data patterns. Streams of data in real-time help the domain researchers analyze real-world events with geographical information [13]. In the stream of text, event occurrence also depends on the arrival of events. In this paper, we are considering the twitter stream of text. Thus, the stream S is an extraction of the tweet T_w in a closed time interval of time, $[t_1, t_2]$ which contains f tex sequence $T_w = (t_{w_1}, t_{w_2}, t_{w_3}, ...t_{w_n})$. A steam sequence S is a set of all the streams collected duringthe interval $S = \{s_1, s_2, s_3, ...s_n\}$. Some part of S such that s_i includes tweets t_{w_i},which may be or may not be linked to other social network entities,we ignored those tweets. In our article, we calculate the burst time of an event directly related to the number of hits and number of users' posts on a particular topic for a specific period T. Thus, any event's burstiness factor can be analyzed when the frequency of the features related to an event E in the text stream S is extensively greater than expected. A threshold value is to be established for analyzing the burst rate of an event. Consider δ is a threshold value, and an event E having a similar or related keyword is occurring again and again in a particular time t_i in time series $T = (t_1, t_2, t_2..........t_n)$. It can reach the value of threshold or cross δ, which is called a bursty event. In other words, it is also known as a trendy event over a particular network.

1.1 Motivation and contribution

The main motive of this research work includes the following:

- This study aims to analyze the trend of an event over a particular time and find out the reactions of the participants and users related to that related event.
- This research article also highlights the event detection methods for social text stream data whose motive is to provide an inclusive view of the recent development in event detection, classification, and burst detection field.
- In this article, for designing and understanding such events' behavior, especially for a critical scenario like real-time text stream, the quality of results depend primarily on the methodologies employed under each phase of event burst detection.

On the other hand, the contribution of the research work includes:

- This research work deals with the real-time stream text to analyze burst behavior of events, which is very useful in social, cultural, political, and other planned or unplanned events cases.
- Usually, data on social media are written with informal intentions, which is a significant challenge for every researcher to clean it. This article has achieved a good record to deal with informal text, which includes misspelling, grammar errors, slang language, irregular abbreviations, mixed languages, and improper sentences.
- Various features of the events have been explored, and they are categorized based on their similarity index using clustering. The diffusion rate is also calculated for features, which leads the event features toward trendy over a network from which a burst behavior can be measured.

The above-described points on which our article contributes are divided into various sections, which are described below.

2 List of notations

Table 1 is defining all the notations which we are using in this article.

3 Related work

Various definitions of event detection, classification, and bursty event are available based on multiple descriptions of context and polarity. Orr et al. [14] context of newswire documents is considered in which authors structured event detection as recognizing “trigger words” and classify events into “refined types.” It is quite challenging to expect trigger words in social text streams due to stream text’s unstructured nature. Identify the features of the unstructured text are quite tricky. Text in social stream contains various symbols (@, #, etc.) [11], short forms (hlo, hru, luv, etc.), which make social text noisy and challenging [15, 16]. Cluster based approaches for event detection and classification task for online incremental clustering in [17] are used. Similarly, in [12, 18] clustering mechanism of event detection is explained in which the similar events are clustered together into a suitable cluster [19]. In [20] with clustering of related events together authors also work to recognize event which is to be clustered. In [21, 22], the authors discussed and explained the most important event detection approaches using unsupervised machine learning and clustering, respectively. C. Aggarwal [23] defined that the clustering stream problem is closely related to determining events. Allan et al. [24] explained and studied new event detection and tracking. They developed a novel approach to single-pass clustering and online adaptive filtering to handle evolving news events. Content based methods using supervised and unsupervised techniques for event detection are present in [25]. Xiaowen Dong et al. [26] use wavelet transform for multi-scale event detection in temporal and spatial scales. The authors also developed a novel approach based on graph-based clustering to compute text similarity. Hilla Becker [27] studied the event identification problem and clusters the event documents with multiple features linked to specific similarity metrics (Table 2).

On the other hand, burstyor trendy terms have their significance on Twitter. Abursty or trendy event also plays a vital role in data/text mining. The evolution and occurrence of similar texts strangely at an unusually high rate for a particular context influence text streams to be a bursty event. Several studies and experiments have been conducted to identify the bursty term from massive text streams. TwitterMonitor [28] is classifying bursty words using greedy methods. It coupled words into a cluster based on their co-occurrence in tweets where every cluster corresponds to an event. In [14, 29, 30], incremental clustering methodologies are explained, and the domain of event evolution is also explored and solved [31]. In incremental approaches, they will be updated on the arrival of new text streams. In [32], authors contribute to detect the bursty events during the US election 2016. They also determined the influence on the growth based on historical data

Table 1 List of notations

Notation	Meaning	Notation	Meaning	Notation	Meaning
"E"	Event	t_α	Initial time	e_f	Value of any feature
"T"	Time	C	Cluster	σ	Standard deviation
$t(e_t)$	Events occurring time	C_{lf}	Life of a cluster	f	Feature
"U"	User	t_α	Creation time of cluster	t_i	Particular time from a time series
ξ	Frequency of a particular event	t_χ	Time when last event feature added	μ	Mean value
S	Stream of text	η	Clusters survival time	δ	Threshold value
T_w	Tweets	Υ	User defined	s_i	Part of stream text
(U, L)	Upper and lower bound	Ω	Value of threshold of various feature of event	w	Weight updating

Table 2 Comparison of related work

Methods	Various approaches and mechanism								
	A1	A2	A3	A4	A5	A6	A7	A8	A9
[9, 18, 21, 23]	●	○	○	○	○	○	○	○	○
[11, 29–31]	○	●	○	○	○	○	○	○	○
[27]	○	○	●	○	○	○	○	○	○
[38, 44]	○	○	○	●	○	○	○	○	○
[22, 25]	○	○	○	○	●	○	○	○	○
[33, 35, 36]	○	○	○	○	○	●	○	○	○
[37]	○	○	○	○	○	○	●	○	○
[39]	○	○	○	○	○	○	○	●	○
[28]	○	○	○	○	○	○	○	○	●
BUrST	●	●	○	○	○	○	○	○	○

●: Satisfied; ○: Unsatisfied; A1: Clustering-based approaches; A2: Online incremental based clustering; A3: Graph-based clustering; A4: Supervised Learning; A5: Unsupervised Learning; A6: Probabilistic based approaches; A7: Query-based event detection; A8: Hierarchical based clustering; A9: Greedy Method

and the result of the election. Probabilistic approaches are also having a significant impact on bursty event detection.

In [33], a parameter-based free probabilistic methodology is designed. Authors claimed that the proposed approach searched the smallest set of bursty features that signify a bursty event. A clustering approach is also used, which groups the collected features and analyzes bursty events' peak times. In [34] also used a probabilistic approach in which authors find a topic of interest and extract patterns from the text stream. A clustering mechanism is applied, which will group all similar words for every time phase, and then applied Kullback–Leibler divergence measure to find out consistent topic over time. Query-based event detection methodology is proposed in [35]. In this case, a tweet is represented by various features in which firstly a number of words and position of the query word in tweet id identified. In the next step, words available in the tweet are analyzed for their codependency and relative ordering. In event classification, the Support Vector Machine classifier is used in [36]. It classifies the tweets into positive and negative classes based on event queries. The author uses this classifier to sense an earthquake and [37] distinguishes non-event words from event words. For this purpose, authors converted words in a string of symbols and then apply hierarchical clustering to produce similar event clusters. In [38] senses bursty event from Twitter, and in [39] online clustering mechanism for topic discovery is explained. In [40, 41] sub-event representation is analyzed in which event time and location are identified by using tf-idf approach. A clustering mechanism is applied, which is based on the self-organizing map. Gigantic growth and evolution of event detection and trend prediction are infrequently stated as burst, which is usually used to describe burst prediction in the text stream. The trend in the text stream is directly associated with the features related to the topic using supervised learning. It gives sharp growth to the features, and this observation can be used to predict trends and popular events in social media [42]. In social media community social users share their opinion in which the content can be extracted from news articles or from real-world social text stream/data sets. In [43] the proposed approach based on real-world dataset BuzzFeed is used for fake news detection. Various features have been used for news classification purpose. Similarly, in [44] a time based trend prediction approach is designed which works on a particular time frame on the based on citation trend. Feature based similarity and semantic based similarity approaches are also very common in social media text analytics and in [45, 46], spam detection approach is designed using machine learning approaches. The authors have designed a methodology which is helpful to analyze the correlation between feature values and emotion values in a particular time series.

4 Problem formulation and designing mechanism

This section describes the problem formulation in bursty event detection, and its designing mechanism is also introduced.

4.1 Problem definition

In this section, we are exploring the various methods of event detection. We have defined the problem statement and few expressions for data pre-processing. Then we explained the three approaches that receive the preprocessed text as input and output the perceived events. Next, we discussed the proposed methodology with essential terminology. The main motive is to intend a plan for the real-time system, and for this purpose, we designed an interface to collect twitter stream text. We carried our experiment based on query-based crawling of the text stream. Our designed interface can collect and analyze the stream generated and posted by numerous Twitter users. To analyze the burstiness of an event, we calculated frequency based on the time directly proportional to interest arguments in a particular time domain. Since the clock cycle of the time domain and for this purpose, we assume that the new event topics can be analyzed or classified when text is crawled in the stream. Clustering methodology is to be implemented, which will group similar events. A threshold value is to be set, identifying and calculating the number of hits by various circumstances. After each elapsed time domain, the approach's output is the directory of multiple classified events in a text stream. A classified event's frequency is checked, an event with a high frequency is classified as a burst.

We assume that the designed model can be appropriate for various real-world situations (natural disasters). Calculating the event's frequency from the threshold function, we can detect its influence in a different time slot. The designed approach is interesting and relevant for short text messages on twitter, text stream, and other short message blogs. In general, Fig. 1 is depicting the various steps involved in burst event detection. This approach is also applicable to detect the burst rate of cultural events, festivals, and elections. Still, it demands more time, such as hours or

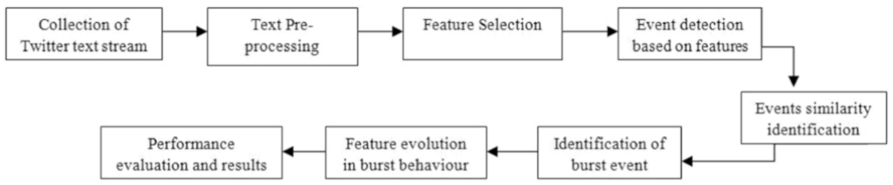
Collection of Twitter text stream
Text Pre-processing
Feature Selection
Event detection based on features
Events similarity identification
Identification of burst event
Feature evolution in burst behaviour
Performance evaluation and results

Fig. 1 Pipeline of event burst detection process

even days, because such discussions go long and have a high impact on users and the location.

4.2 Designing methodology

Text in the form if the stream is extracted and it is normalized before further processing. As the crawled text will be noisy and consist of various unwanted symbols, abbreviations, and short forms. Thus for cleaning the stream of text, normalization is to be implemented. For this purpose, we used the methodology described in [15]. In text pre-processing, the neighboring binding of the slang words existing in text corpus with diverse senses (coexisting) so far based on bigram and trigrams language models is calculated. With both language models, the prospective sense binding vector is collected with the systematic words (pair and triplet ordered pair), and their related probabilistic weight is also calculated. In the next step, the author [15] has decided the significance of the unidentified word. Word2Net dictionary is also used to identify the words which are OOV (Out of Vocabulary). After this process, clustering is implemented as the data are continuously coming, and it is quite challenging to analyze its features to combine. Thus, to obtain and handle this challenge, we use an online clustering mechanism to buildup information comparative to the text stream that is already classified and normalized to assign clusters accordingly. To make this concept clearer, we describe few definitions that will provide a crystal clear situation.

4.2.1 Tweet and stream of a tweet

Tweet T_w can be in any form out of seven major categories: Regular, Image, Video, Location, @mention, Re-tweet, and Poll tweets [47], and can be represented as $T_w = \{R_T, I_T, V_T, Loc_T, Tag_T, R_T, P_T\}$. T_w Facilitates the users to share, deliver, and interpret 140 characters post [48] whereas, S is a set of all the streams collected during the interval $S = \{s_1, s_2, s_3, ...s_n\}$ in a closed interval of time, $[t_1, t_2]$ where the initial time is t_α where $T_w = (t_{w_1}, t_{w_2}, t_{w_3}, ...t_{w_n})$ is the representation of a sequence of the text stream.

In this way, we originate the research issue of detecting events in real-time text stream and cluster together the interesting pattern using cluster mechanism. The flow of text from the stream will be in a continuous fashion. It's the cluster's responsibility C to manage the traffic, assign the cluster to every upcoming event, and update its value C. During the crawling of text from the twitter text stream S, the tweet t_{wi} will connect the most related cluster created before the similarity between tweets' texts. The available cluster is greater than the threshold δ otherwise new cluster C_{new} will be made.

From this behavior of tweets, bursty nature can be calculated because the cluster has maximum hits of similar events quickly. This dynamic nature of the cluster makes our approach more efficient. A cluster C will be dynamic $C_{dynamic}$ if it grows with time T, whereas it does not update its value with the change in time in a static cluster.

4.2.2 Cluster and life length of a cluster

The cluster concept in the text stream is necessary to categorize the similar contents in the summarized form. For this purpose, text streams,$S$is distributed among clusters C, which is defined as $C = (C_1, C_2, C_3, ...C_n)$ where, $C = (C_1, C_2, C_3, ...C_n) \in S$ such that $S \cup (C_1, C_2, C_3, ...C_n)$. The tweets from the text stream s_i belong to C_j a cluster based on similarity at a time t_j where $C_j \subset C$. Consider C_{lf} representing the lifetime of a cluster, where t_α the creation time of cluster t_χ is when the last event keyword is entered in a cluster and η is a survival time function for a cluster after creation (even no tweet added after creation time) of a cluster. Whereas,Υ is a user-defined factor, which indicates the dynamic nature of the cluster. The whole scenario is represented in Eq. 2.

$$\eta + 2^{\Upsilon} \times (t_\chi - t_\alpha) \tag{2}$$

4.2.3 Decomposition rate of a cluster

It is usually represented as, $\tau = \frac{1}{C_{lf}}$. We check the dynamic activity of a cluster during event burstiness as it receives upcoming tweets from the text stream. Usually, a high frequency of a particular event in a short time period is considered a bursty event. Consider $t_\vartheta = (\eta + \Upsilon)/\vartheta$ being the duration of time span. This factor will help us evaluate the busrtiness by checking it at the end of the time span. ϑ is a user-defined parameter directly proportional to the $(\eta + \Upsilon)$ indicating that if the value of higher ϑ, shorter will be the time to check the bursty events.

4.2.4 Similarity measures of streaming text and existing tweets

Similarity of the text stream tweet can be computed between the new tweets coming from the stream window and the tweets existing in various clusters using $sim_{(t_w, C)}$. We are considering the various features of the tweet in which we are considering the location, time, type of tweet, and user, i.e., $(t_{w_{(loc)}}, t_{w_{(time)}}, t_{w_{(rt_w)}}, t_{w_u})$. Consequently, we use the standardization method (standard score) based on features characterized. From this standardization, we can predict the number of users that sent the tweets to the dedicated cluster C, various locations from where the tweet was posted, the total amount of tweets posted in a particular time span, and its tweet type, i.e., either re-tweeted or not. A higher standard store in any values of $t_{w_{(loc)}}, t_{w_{(time)}}, t_{w_{(rt_w)}}, t_{w_u}$ is a clear signification that there has been an unusual intensification of tweets. Suppose e_f symbolize the value of any feature in a particular time span, then the standard score can be described as follows:

$$Z(e_f) = \frac{f - \mu(f)}{\sigma(f)} \tag{3}$$

Here in Eq. 3, μ is a mean value, e_f is the feature associated with an event, f is a feature that is computed, and σ the standard deviation. Whereas we will compute the mean and standard deviation of the event's feature f and use the way described in [49] figured out as given in Eq. 4.

$$\mu_{\wp} = \mu_{\wp-1} + \frac{1}{\wp}((e_f)_{\wp} - \mu_{\wp-1}) \tag{4}$$

$$S_{\wp} = \mu_{\wp-1} + \frac{1}{\wp}((e_f)_{\wp} - \mu_{\wp-1})((e_f)_{\wp} - \mu_{\wp}) \tag{5}$$

$$\sigma_{\wp} = \sqrt{\frac{S_{\wp}}{\wp}} \tag{6}$$

Thus, to compute the Z-Score of the feature of events, we relate Eqs. (4) and (5) for the incremental mean and variance estimate.

4.2.5 Optimization of events stream and threshold

In this section, we describe the machine learning-based approach to optimize the stream of text with a threshold for detecting events. The analysis approach is based on a given stream of text $S = \{s_1, s_2, s_3, ...s_n\}$ and the tweets entities in a cluster C based on a similarity function $sim_{(t_w,C)}$. The main objective of optimization for text stream is to map data to precise space to obtain reliable representation [50, 51]. When enter based on cluster threshold metric, any event from the twitter stream will be calculated for every cluster for a particular time span. For this research article, we measure the threshold based on time series in which we will check the threshold value, and the event which is approaching the value of the threshold will be declared as a bursty event. We are also taking care that in a particular scenario, sometimes the events travel a long time. We can say that the survival time of an event can be extended, which affects the burstiness. By taking care of this situation, we are considering this based on time series. An event approaching the threshold in less time will be bursty, and others will be weeded out.

Algorithm 1: Procedure to calculate thresholds	
Input:	$\forall$ Cluster C threshold value for time t is computed δ at different time span $T=(t_1,t_2,t_2.........t_n)$ $\forall$ features of an event related tweet such as $(t_{w_{(loc)}},t_{w_{(time)}},t_{w_{(rt_w)}},t_{w_u})$
Output:	***Threshold*** $\delta=\{\delta_1,\delta_2,\delta_3,.........\delta_n\}$
Step 1	**Apply learning algorithm to** $\mathrm{E}(t_{w_{(loc)}},t_{w_{(time)}},t_{w_{(rt_w)}},t_{w_u})$, $S=\{s_1,s_2,s_3,.......s_n\}$ where E is a set of event including its features and S is a set of twitter stream to obtain the lower and upper bound for every feature which is calculated in next step.
Step 2	$(\mathrm{M}_{\max},\mathrm{M}_{\min})\rightarrow(\mathrm{M}_{\max_i},\mathrm{M}_{\min_i})\forall(t_{w_{(loc)}},t_{w_{(time)}},t_{w_{(rt_w)}},t_{w_u})$ where (U,L) are upper and lower bound respectively.
Step 3	Calculate the threshold for every feature of Clusters at time t_i such as: $\delta_i\leftarrow[(t_{w_{(loc)}})_{C_i}]\leftarrow t_i$ $\delta_i\leftarrow(t_{w_{(time)}})_{C_i}\leftarrow t_i$ $\delta_i\leftarrow(t_{w_{(rt_w)}})_{C_i}\leftarrow t_i$ $\delta_i\leftarrow(t_{w_u})_{C_i}\leftarrow t_i$ $\therefore\forall\delta\bigcup S$ Where, S is a set of all features of events at different time span.
Step 4	Similarly compute threshold for all other clusters at different time span which is represented as: $\delta\leftarrow\Omega_{\mathrm{M}}$, where Ω is giving value of threshold for various features of an event.

After calculating the threshold for various features of the event, a cluster is declared bursty if any of the features $(t_{w_{(loc)}},t_{w_{(time)}},t_{w_{(rt_w)}},t_{w_u})$ defined in step 3 are more significant than the other feature values of the clusters. The mechanism of online clustering is described in the next section with its proper functioning. It also works for assigning a similar cluster to the tweet based on the similarity function.

Algorithm 2: BUrST

Initialization:

```
1:  Input: A continuous stream of tweets Text stream $T_w = (t_{w_1}, t_{w_2}, t_{w_3}, .........t_{w_n})$, a
        threshold $\delta$, Cluster survival time $\eta$, $t_\alpha$ is a creation time of cluster, $t_\chi$
        time when cluster will not accept more event i.e. the time when last event
        entered, $t_\vartheta$ is the time span, and $C_{lf}$ is the life or survival life of a cluster.
    Output: Active cluster $C$ with collection of a set for bursty event $\mathrm{B}$
2:  Begin
    $t_\vartheta = \frac{(C_{lf} + h)}{\vartheta}$
    $C \leftarrow \phi$
    $\mathrm{B} \leftarrow \phi$
    i=1
    $SO_1 \leftarrow t_{w_1}$ ,Creation of social Object for tweet
    $C_1 \leftarrow SO_1$, Creation of cluster for social object of text stream where other clusters
    will be created in same way where $C = \{C_1, C_2, C_3.........C_n\} \ni C \leftarrow C \cup C_1$
3:  While (not end of stream)Compute indicator dataset of text stream corresponding
    to $i$
    $i \rightarrow i+1$
    Obtain the next tweet from stream $t_{w_2}$
    Apply step 2 for the creation of social object for the new coming tweets
4:  for each cluster $C_k = \{C_1, C_2, C_3.........C_n\}$
    if IsDynamic( $C_k$ ) then Similarity of each cluster is computed as:
    $(SO_i, C_k) \leftarrow ComputeSimilarity(SO_i, C_k)$
                 else      $C \leftarrow C - C_k$
              END
    END
5:  Computing bursty event from threshold
    if      $sim_{(t_w, C)}(SO_i, C_k) \geq \delta$, then check burst as:
    $CheckBurst(C_k)$ then
    $\mathrm{B} = \mathrm{B} \cup C_c$,
    End if
    Computenew threshold value by accepting new coming stream of text from
    cluster.
6:  Else
    $C_x \leftarrow CreateCluster_{(SO_x)}$ where $C = C \cup C_x$
    END if
    END while
    End
```

5 Proposed online clustering mechanism

5.1 Procedure

Proposed methodology BUrST is an augmentation process. The primary purpose is to process all the incoming tweets of streams and classify the tweets according to their similarity. It also groups all tweets in a cluster which represents an event. Further, it recognizes the bursty event based on the burst rate measured with the threshold value δ.Identifying the bursty nature of cluster time plays a vital role because this methodology collects tweets $T_w = (t_{w_1}, t_{w_2}, t_{w_3}, ...t_{w_n})$. The cluster C_i will be generated based on incoming tweets from the stream of text. In the first step, any tweet enters, the new cluster will be generated, and identity is assigned to a particular cluster based on the event type. Similarly, in the next phase, the new cluster will be generated, but before developing a new cluster, the similarity of the upcoming event's tweets is to be analyzed using the similarity function $sim_{(t_w,C)}$. A social object SO_1 is created for every dedicated tweet by setting their equivalent frequencies. Further, the event similarity is identified, and it helps to estimate the bursty nature of the event. On the other hand, if a new tweet t_{w_k} arrives at the time $\boldsymbol{t_k}$, it indicates that it is not an end of a stream and a social object is to be created for fresh coming tweets for analysis. In the next step, the static clusters are eliminated from the group of clusters. Based on the similarity function as defined in the BUrST, if similarity elapsed $sim_{(t_w,C)}(SO_i, C_k) \geq \delta$, the function $CheckBurst(C_k)$ is executed. These bursty events were then put into a function B that indicated the discovered bursts. In the above scenario, it is very clear that burstiness can be found, but it also demands updating the different time slots' threshold values. Therefore, an updating function must be updated every text stream's value after assigning a cluster to a particular event. For this purpose, we are using the stochastic gradient descent method using a learning rate w is used for weight updating. From BUrSt, we have achieved our target to find threshold value δ. Now the difference between the old and new frequency is also calculated (Eq. 7).

$$Q(\delta)_{c_i} = \frac{i}{n} \sum_{i=1}^{n} Q_i(\delta)_{c_i} \tag{7}$$

Here in equation 7, δ is a parameter to learn, and it describes the frequency of events in a cluster c_i. The function Q is directly associated i^{th} with observation of cluster C, which is associated with its frequency rate. Therefore, from Eq. 7, we will calculate every frequency δ and give the event's updated value burst.

5.2 Similarity measure

In any clustering approach, cluster groups all similar types of objects together. The algorithm's efficiency and success are also directly related to the similarity calculated by the cluster's similarity function. Based on that, during experimentation, we found that the tweets, including the hashtag (#) or @ symbol, usually have more priority and are related to the re-tweet or addressing hype topic, person, place, or something. Sometimes in computing the similarity of tweets, the lexicon may be changed, and ambiguity can exist, leading to creating a new cluster. Thus, due to this issue, we have to analyze the tweets having multiple words with the same meaning to compute the similarity efficiently without leading to a new topic or event. A tree data structure represents the event-related keywords in which one root word related to event will combine all other event related words together in a single tree to count the occurrence of the same event word. Dependency parsing is a lightweight syntactic formalism that relies on lexical relationships between event words. Due to this reason, the turbo parser is used as it can extract the events' non-local feature and perform approximate inference to speed-up the event detection task. The architecture used to represent the whole scenario is layered (n-tier architecture), where each layer has individual responsibilities and ways to manage dependencies among event features.

5.3 Pseudocode for evaluating burstiness

Pseudo code for evaluating an event's burst behavior is described, which is working on the various features captured by an event at the time T. Based on features cluster address the events, the burst event cluster can be gained as explained below by following the given procedure.

```
Input   Cluster $C$ with Tweet $T_w$ including various features as:
        $T_w \rightarrow T_w = (t_{w_1}, t_{w_2}, t_{w_3}, \ldots\ldots\ldots t_{w_n}) \Rightarrow (t_{w_{(loc)}}, t_{w_{(time)}}, t_{w_{(rt_w)}}, t_{w_u})$ at time $T$.
Output  Declaration of bursty Cluster
Begin   $Z_{e_f}(T\ell.t_{w_{(loc)}}) = \frac{T\ell.t_{w_{(loc)}} - \mu(T\ell.t_{w_{(loc)}})}{\sigma(T\ell.t_{w_{(loc)}})}$
        $Z_{e_f}(Tt.t_{w_{(time)}}) = \frac{Tt.t_{w_{(time)}} - \mu(Tt.t_{w_{(time)}})}{\sigma(Tt.t_{w_{(time)}})}$
        $Z_{e_f}(Tr.t_{w_{(rt_w)}}) = \frac{Tr.t_{w_{(rt_w)}} - \mu(Tr.t_{w_{(rt_w)}})}{\sigma(Tr.t_{w_{(rt_w)}})}$
        $Z_{e_f}(Tu.t_{w_u}) = \frac{Tu.t_{w_u} - \mu(Tu.t_{w_u})}{\sigma(Tu.t_{w_u})}$
if      $Z_{e_f}(T\ell.t_{w_{(loc)}}) \geq \delta \,||\, Z_{e_f}(Tt.t_{w_{(time)}}) \geq \delta \,||\, Z_{e_f}(Tr.t_{w_{(rt_w)}}) \geq \delta \,||\, Z_{e_f}(Tu.t_{w_u})$
then    C is a cluster which is bursty.
        End
end
```

Table 3 Depiction of data sets before and after normalization process including features

Data Set	Temporal Coverage (in different time span)	No. of tweets (before normalization	No. of tweets (after normalization	Number of Features
FA CUP	12 h	188,564	437,453	13
Enron mail stream	24 h	485,619	459,411	27
US election	36 h	164,761	152,412	32
Canada Election 2019	36 h	982,123	945,512	240

6 Experimental study

The proposed methodology is evaluated by experiments that are implemented to reveal the effectiveness and efficiency. We compare our methodology to compare it with three states-of-the-art approaches using three different data sets. In this section, we discuss the existing approaches used in our experiment, which are explained in [52], such as LDA, graph-based, and bn-gram. We also explained the data sets used in this research: Twitter text stream, Enron mail stream [53], and US Elections [52].

6.1 C. latent dirichlet allocation (LDA)

Probability based event detection methods are famous for addressing the topic detection problems in textual datasets. In other words, we can say that the topic model is a Bayesian model. Each document is associated with a probability sharing over topics and turn allocation over topics. LDA is a widespread and widely used model due to which reason, and we select it as a baseline to evaluate our approach [54]. In LDA, each document is referred to as a bag of words. These words/terms are the only examined words in the model. Bayesian inference is used to guess and estimate the distribution of document and term distribution per topic hidden. LDA necessitates an anticipated amount of topics *k*.

6.2 Document-pivot topic detection (Doc-p)

Another approach that we have used is a document-pivot approach. As in [55] LSH is used to extract the nearest neighbor in the document. LSH is adapted to rapidly retrieve the nearest neighbor of a document and speed up the clustering mechanism. The pre-processing is to be performed on the document, which is already explained in [15]. In this process, the steps involved are performing online clustering on real-time event posts, finding the best matching tweet from the posted tweets to detect event similarity, assigning cluster based on similarity, and compute the similarity score. The method to compute the similarity score [52] is given below (Eq. 8).

$$score_c = \sum_{i=1}^{|Doc_c|} \sum_{j=1}^{|words_i|} \exp(-p(w_{ij})) \tag{8}$$

6.3 Graph-based feature-pivot topic discovery

We used another approach to extract unique features in the collect documents. Structural Clustering Algorithm for Networks (SCAN) [56] is used by this approach to cluster the documents. This SCAN approach perceives nodes' communities and provides a detail of the center, each of which may be attached to a multiple set of communities. In this way, a network is generated where each community exchanges information to the related community. It also provides an effective way to a specific link between topics. Similar words can be cluster together based on similarity, and the importance of the words can also be computed by considering the probability of words. As in [57], the probability of a word is computed as $p(w|textstream)$. Generally, this approach includes four folds which are selection, linking, clustering, and clustering enrichment.

6.4 BN-grams method

This method contains three steps: the calculation of DF-IDF calculation in the first step. The second step performs the n-gram clustering in which similar keywords from the twitter stream are combined. In the third step, topic ranking is performed. Each tweet is normalized/pre-processed for this purpose; we use a similar approach again as described in [15]. In our research work, we consider two types of aggregation such as time and topic [58]. Furthermore, LSH is implemented on the collected tweets in a particular time slot based on tweet similarity.

7 Evaluation mechanism and dataset: implementation detail

The outline of evaluation includes multiple stages, which are explained below:

- *Data collection:* we extract the text from the Twitter API for significant events such as Enron mail stream, US elections, and FA Cup. Various challenges occurred during the collection of text streams, and to overcome these challenges, we designed a real-time system to consume the Twitter API. To create an evaluation dataset, we used three types of widely used data sets from which various entities and other features required for event classification are extracted from these input tweets. We collect the Tweets related to these data sets, and accordingly, keywords and hashtags are applied at different time slots of stream-

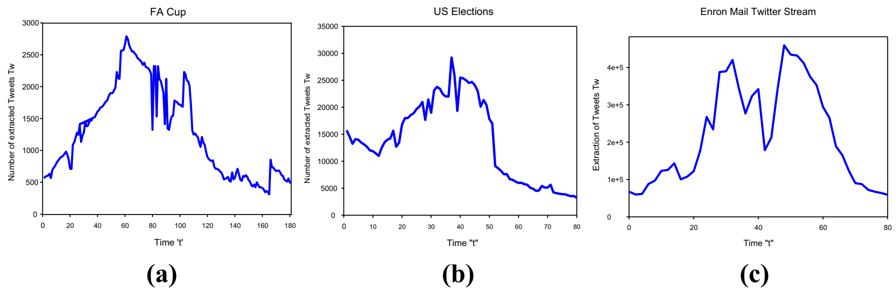

Fig. 2 Process of collecting Tweets from Twitter: FA Cup Stream (**a**), US Election text stream (**b**), Enron mail text stream (**c**)

ing text for extracting data. The streaming of extracting data from Twitter related to events is extracted minute by a minute basis. Relying on the Twitter streaming mechanism, we execute it in an online scheme that can account for the scale and fluctuations of the Twitter stream. The extraction process was divided into various time slots to collect an average amount of tweets according to the aimed event's character. The use of timestamp information in the text stream data set can be converted into streams [8]. In this stream of tweets, each object contained text and the sender and receiver information, including network structure. To understand the email network structure compared with tweets, a tweet with a single sender and multiple receivers communicate. Further, for each cluster related to the event theme, we take all the entities from every point in time and produce one set of event entities based on different features associated with an event. If the feature value of an event matches with the theme of an event (using step 4 of algo1), then a score will be added into a particular event cluster, including event ID and title) otherwise ignored. To handle multiple events features simultaneously, different IDs and titles are assigned for each of them. Table 3 is describing some examples of event related data in different scenarios.

- *Mining the ground reality:* in every text corpus or stream of text, many essential matters are concealed. Therefore, understanding the reality of the awe-inspiring text amount of attempt would be entailed to extract features manually. In other words, we can say that we cannot rely totally on the extracted features from a stream or the features visible in data sets. Every topic, the ground reality, contains multiple keywords on a particular time slot in which features and themes appeared in conventional reports.
- *Theme detection:* every text stream should be based on a theme to be evaluated on each time slot. Every stream of text is divided into various time slots, and we merely consider the text from the stream as input with ground reality.
- *Evaluation of theme detection output with ground reality:* self-detection of a theme in a stream of text is evaluated using Theme Recall, Keyword Precision, and Keyword Recall. The comparison is to be computed at the top *n* extracted topic in a stream.

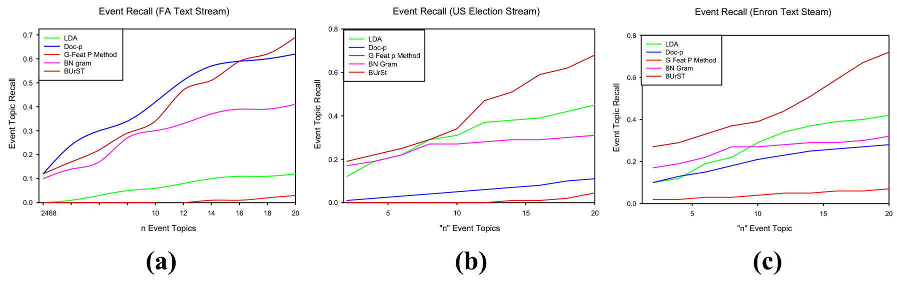

Fig. 3 FA CUP—Event recall (**a**), US Election- Event Recall (**b**), and Enron Mail Stream-Event Recall (**c**) of *BUrST*algorithmw.r.t. existing methodologies

Table 4 Comparison of BUrST (proposed approach) with existing approaches

Method	Event topic recall			Event keyword recall			Event keyword precision		
	FA CUP	US elections	Enron mail stream	FA CUP	US elections	Enron mail stream	FA CUP	US elections	Enron mail stream
LDA	0.12	0.45	0.42	0	0.57	0.52	0.22	0.32	0.61
Doc-p	0.62	0.11	0.28	0.57	0.47	0.31	0.31	0.29	0.34
G feat P methcxod	0.02	0.04	0.07	0.63	0.55	0.22	0.40	0.39	0.44
BN-Gram	0.41	0.31	0.32	0.64	0.44	0.17	0.34	0.46	0.57
BUrST	0.69	0.68	0.72	0.68	0.59	0.76	0.61	0.58	0.72

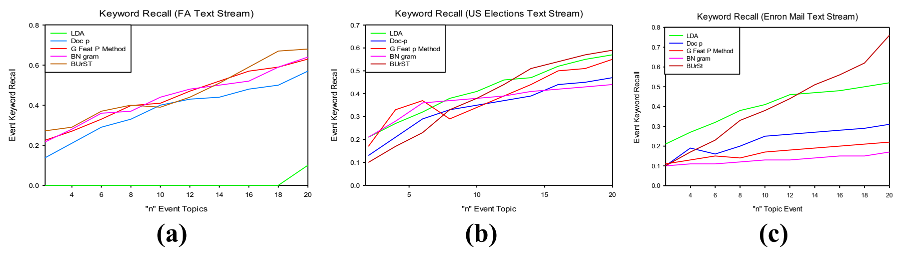

Fig. 4 FA CUP—Keyword recall (**a**), US Election-Keyword Recall (**b**), and Enron Mail Stream- Keyword Recall (**c**) of *BUrST*algorithmw.r.t. existing methodologiesf

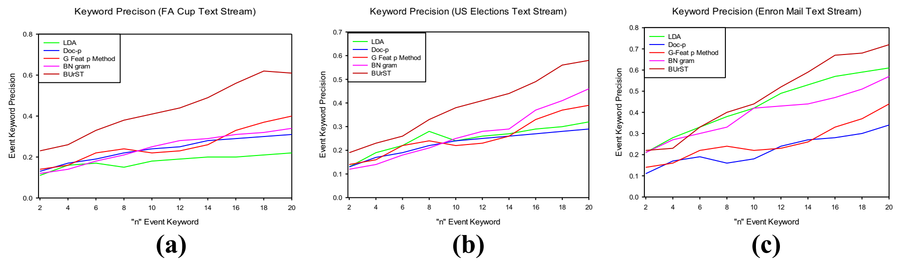

Fig. 5 FA CUP—Keyword Precision (**a**), US Election- Keyword Precision (**b**), and Enron Mail Stream-Keyword Precision (**c**) of *BUrST*algorithm w.r.t. existing methodologies

Table 5 Processing time of BUrST (proposed approach) with existing approaches

Method	Processing time in seconds		
	FA CUP	US Elections	Enron Mail Stream
LDA	0.76	0.96	1.11
Doc-p	0.99	0.82	0.87
G Feat P Method	0.67	0.47	0.89
BN-Gram	0.65	0.43	0.92
BUrST	0.59	0.42	0.83

7.1 Data sets

In this article, the following events are used to find the burst rate in the real-time scenario. These events show how deeply the burst nature of events impacts the social uses and the whole world. Event analysis has changed itself to become the brain of every social user. Event burst analysis helps to know the impact on various users as we have seen that there is no any evidence in which the burst nature of the considered events has been analyzed. We are choosing a combination of 2012 events which are the FA Cup Final and US Elections. Whereas the two events are relatively new in which a detailed analysis of the Canada 2019 election is described. Table 3 is describing the detail of tweets before and after the normalization process.

7.1.1 FA cup final

As we know, the FA Cup or Football Association challenge cup is the world's mature association. The first competition was held in 1871 and is the chief competition in English football [2, 52]. Chelsea and Liverpool were the two finalist teams in 2012 between which the match was organized, and it was the seventh time when Chelsea won the FA Cup. Data were extracted using official hashtags (#tags) for events using team names and players' names. The ground reality included 13 topics such as start, middle, and end of the match time, bookings, and goals (kaggle.com).

7.1.2 US elections

This dataset provides information regarding the presidential election of 2012. Various types of keywords are used to extract the text. The ground reality includes 64 topics. Multiple hashtags (#tags) are used for crawling data such as presidential race in a particular state, result, senate race result, and victory speech of Obama's [52]. Elections of the US magnetized enormously high levels of movement on Twitter, and due to that reason, numerous users tweet on the same topic and provide massive data to extract useful patterns. For more exploration, we can refer (kaggle.com) to see the various unique features for reference.

7.1.3 Enron mail stream

With the time stamp information of the emails, the data set can be converted into streams [8]. In this email stream, each object contained text and the sender and receiver information, including network structure. To understand the structure of email network comparison with twitter tweets, a tweet with a single sender and multiple receivers communicates. In this data stream total of 617,432 are present, from which few are eliminated due to invalid users, senders and receivers. In this process, few emails having duplicated entry, and having invalid text are also filtered out. The Enron email data stream contained a total of 617,432emails. We eliminated emails

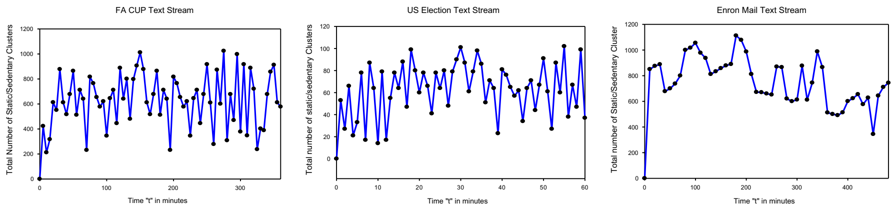

Fig. 6 Inspection of a total number of clusters that turn into static/sedentary after a group of tweets based on cluster capacity in the various text streams

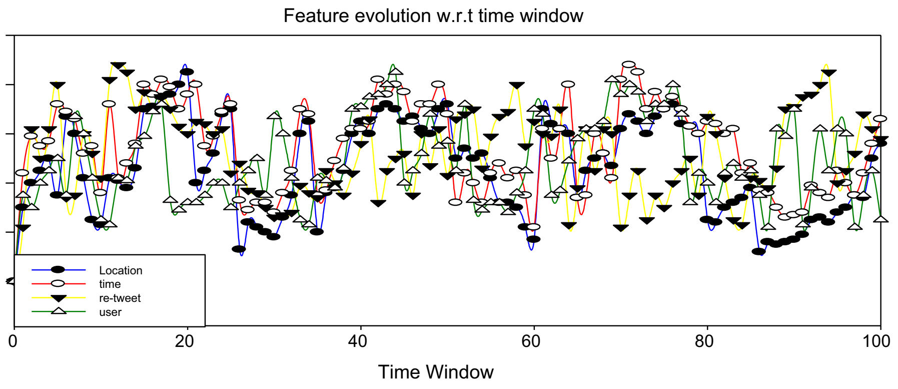

Fig. 7 Feature Evolution for the event "Canada Election 2019"

that did not have valid sender and receiver email identifiers. After applying normalization, we got 459,411 emails with accurate information. (https://www.kaggle.com/wcukierski/enron-email-dataset).

Canada election 2019: This data set provides the information related to the Canadian election of 2019. Various features and entities are extracted to explore the features of this data set. Total 640 unique features are identified, which are described in Table 3. Numerous users' information types can be extracted and studied from the data set (https://www.kaggle.com/smid80/canadian-federal-election-results-timeseries) to extract useful, informative patterns. The tweets are extracted in different time stamps, and the tweets are extracted into various time spans (36 h). A total number of 945,512 tweets is used for the exploration of various features.

From the above Fig. 2, it is evident that the data collection mechanism started many days before finding useful patterns of events. The peaks in Fig. 2 indicate the maximum traffic of tweets during extraction. The topics of the event are considered by various durations, which are divided into different time slots. For experiment purposes in total, we use 437,453 tweets for FA Cup, 152,412 for US elections, and 459,411 emails with valid information.

7.2 Estimation of determining event: performance analysis

In this part, we described and evaluated the capability of our proposed approach BUrST to extract and detect events from the Twitter stream as described in the above sections. We choose LDA, Doc-p, G-Feat p method and BN gram to check our approach's performance. In the first stage, we evaluate our approach by comparing them based on detected events. Various sets of parameters have been fixed for performance checking, and we maintain a few for evaluation. BUrST will accept the input, which also includes another parameter which are *text* stream $T_w = (t_{w_1}, t_{w_2}, t_{w_3}, ...t_{w_n})$, a threshold δ, Cluster survival time η, t_α is a creation time of cluster, t_χ time when cluster will not accept more event, i.e., the time when the last event entered, t_ϑ is the time span, and C_{lf} is the life or survival life

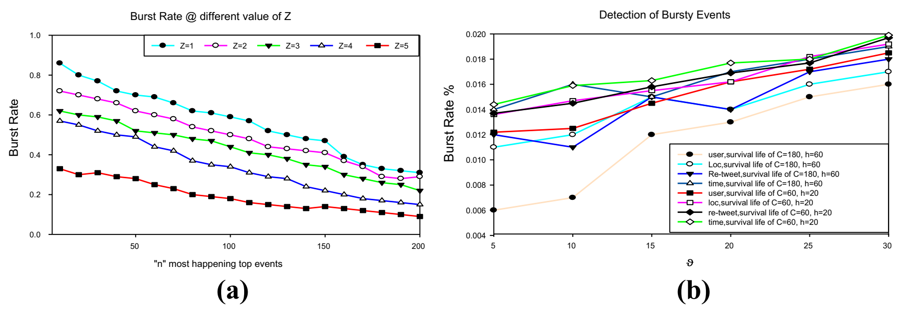

Fig. 8 Bursty Rate (**a**) w.r.t top N events for different standard score values *and* Burst Rate (**b**) w.r.t "ϑ "for different values of C_{lf} and different features

of a cluster. The data set size is not fixed, and it varies and depends on many parameter configurations. For this purpose, we compute the proposed approach's performance by increasing the number of events with the time change. The experimental results described in Fig. 3 indicate that if the value of input increases with the increase of time, then the similarity index, i.e., a threshold δ, the recall gets poorer. We found the evaluation with correlated approaches by showing in Table 4 the precision and recallmetrics in fixed number N of events. Specifically, for the FA Cup dataset, the number of events is set to $N=2$, and for the US Election dataset, $N=10$. These values are those chosen in [47]. For the Enron Mail text stream, we set $N=20$ (Figs. 4 and 5).

From results, it is examined that at $N=2$, the proposed approach BUrST produced the best result in event recall. We observe that for the FA Cup dataset at

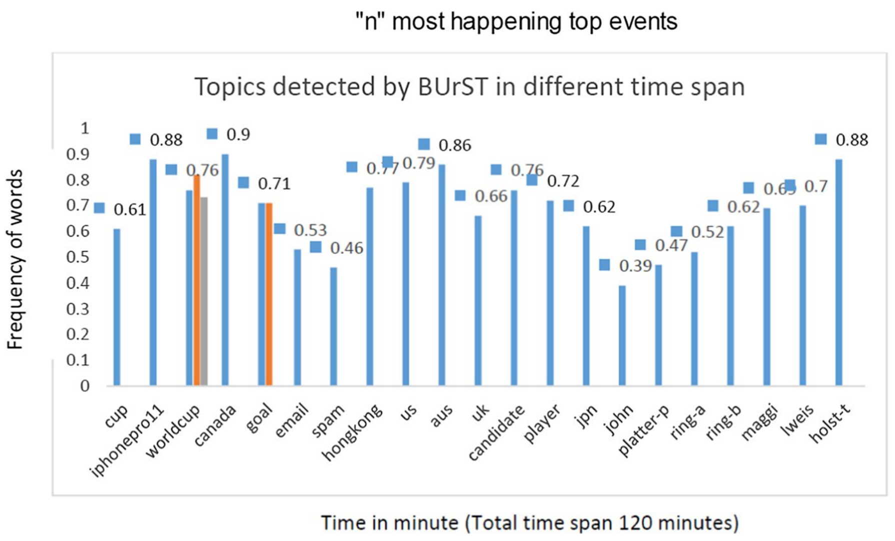

Fig. 9 Bursty topics detected by *BUrST (proposed approach)* in 120 mins

$N = 2$, the proposed approach *BUrST* achieved the best event topic recall. We also observed that BUrST detects topics very nicely and describes events keywords from the topics very clearly as we gain good value in keyword recall (0.76). Similarly, BUrST is giving a good performance for the US Elections text stream and the Enron mail text stream. In both cases, BUrST achieved the uppermost event recall followed by LDA. The poor performance was acquired by the G Feat P method (Graph-based) in all text streams. Every method reveals the particular type of target event in the different text stream. FA cup focused on specific keywords directly related to the match and focus on goals in a short period of time. In contrast, in the US election text stream, the whole picture is different because it includes multiple keywords, which create problem to capture the ground truth. Moreover, in the Enron mail stream, it's complex to filter the event related keywords because it includes different arguments, which are very complex to observe.

Therefore, from the result, we can say that BUrST gives good results in different situations, and its performance with the comparison of others is quite good even in complicated situations such as the Enron Mail text stream. Thus, we can say that BUrSTcan extract more topics than the existing methods more appropriately and efficiently. It shows the highest precision rate than existing approaches, which is a good indication with less processing time (Table 4). The computation time of the proposed approach is also measured against existing methods, and we found that the BUrST is performing well than the existing methods. Table 5 is indicating the processing time of various approaches using various data sets.

In the next step of BUrST, we explained the functioning of clusters. For this purpose, two states of clusters are considered dynamic and static states, as explained in Sect. 4.1. Therefore, to close this section, we want to show the benefits of the online clustering mechanism that becomes dynamic when observing new tweets in the text stream. This clustering mechanism is faster than other clustering mechanisms. In our implementation, we observe that the clusters that become static after a group of similar tweets of 1000 are used for further processing. It also indicates that when cluster becomes static, it is not more alive to accept event keywords from the text stream. Continue functioning of a cluster in this type of situation would only slow down the process. Figure 6 shows the dynamic and static nature of clusters in the different time slots. For this purpose, we have divided the time into a different time slot and compute the dynamic clusters in two ways. Firstly, we analyze the dynamic cluster in a particular time slot, whereas secondly, we calculate the total cluster being active in total time.

8 Evaluation of bursty event

The above section explains the method of cluster creation with their static/sedentary time. Now, in this section, we will compute the bursty event with the proposed algorithm BUrST. For this purpose, first, we will extract the features of events to describe the plentiful description of multiple events.

A cluster C with Tweet T_w including various features as: $T_w \rightarrow T_w = (t_{w_1}, t_{w_2}, t_{w_3}, \ldots\ldots\ldots t_{w_n}) \Rightarrow (t_{w_{(loc)}}, t_{w_{(time)}}, t_{w_{(rt_w)}}, t_{w_u})$ at a time T is available. Four features $(t_{w_{(loc)}}, t_{w_{(time)}}, t_{w_{(rt_w)}}, t_{w_u})$ are considered to understand the evolution of any event. At each time span, the value of features is updated as described in 4.2 and in event features e_f symbolizes the value of any feature in a particular time span, and the standard score can be described as described in Eq. 3, whereas a detailed description is given in Sect. 4.2, which updates each feature's value if an event. Based on the standard score, we can declare that the cluster is bursty.

Figure 7 is describing the evolving features of the Canadian election 2019. For extracting features, we explore the various values from the text stream and excavate the useful patterns. The main keywords for the evolution of an event are election identification (election_id) which describes the year of elections and the election number. On this basis, we extract tweets of 2019 polls. We also extract the information based on the region where each region has a unique value and found 640 unique values. Polling station numbers and names are also considered for identifying the candidate list and extracting the information of those winning from a particular polling station. Each polling station has a unique value.

Furthermore, the candidate's name and candidate's poll vote count are the main key factors that help the users post tweets according to their favorite candidate. From this, we can identify the user's most favorite candidate. Kaggle also explained the similar parameters on the website, whose main motive is to describe the various information related to polling. During the extraction, few of the cluster does become bursty whereas few turn into bursty numerous times. This is when the clusters are engaged with Canada polling 2019 results and the most significant events related to the same. Thus, to focus on the main event, Fig. 7 depicts cluster evolution in stipulations of bursty features. We have seen that based on candidate name and location,the cluster becomes bursty several times. Every feature's zig-zag motion is visible in Fig. 7, from which the trend can be fluctuating in a discontinuous manner. Such type of scenario affects the $Z(e_f)$ standard score. Thus, every time span event topic becomes bursty according to their various features, which are visible in Fig. 7.

Moreover, to check the bursty nature of an event, we evaluate the standard score set and identify the burst's nature. We found that burst rate moderates as the number N of top detected events boost from this experiment. We also set the different values of the standard score for different candidates and location considerations for this purpose. The threshold value is also matched with the standard event score to declare the bursty event keyword. After every three minutes, the cluster's burstiness is evaluated.

Now, further, we analyzed the burst rate of an event based on various features Fig. 8b. It is observed that the trend and burstiness are the same for the combination of values of C_{lf}, i.e., the cluster survival life and h for all the four features which are $(t_{w_{(loc)}}, t_{w_{(time)}}, t_{w_{(rt_w)}}, t_{w_u})$. In Fig. 8a, we implemented and checked the performance of top N events for various values of the standard store, and we found that with the increase in standard score threshold rate, there is a decline in the fraction of busty clusters. We observed that the standard score value plays a vital role in identifying burst events and normal unusual events.

Therefore, from all the above discussion, it is observed that for all the approaches, the results obtained from BUrST are the best among all other machine learning approaches and various data sets. We use these data sets due to their popularity and continue updating the ground-level information. In the latest machine learning approaches, we assume that our proposed approach (BUrST) will perform well to handle the diverse and sparse natured data efficiently. The sparse and diverse nature of text stream data is expected due to greater temporal coverage in event topics/features. Therefore, in advanced machine learning mechanisms, it is assumed that BUrST will outperform to sense the burst nature of the event in various kinds of events, as we have seen in the above results (Fig. 9).

The proposed approach's limitation relies on the ontology between the various events in the social text stream. As we have seen, in burst event detection, domain-specific relationship modeling between various event features is required. Therefore, it is necessary to update the relationship with the change of time, including an actual scenario from time to time, reflecting the event's current reality in real-time.

9 Case study—the 117th United States congress (US presidential inauguration)

The 46th US presidential inauguration ceremony is one of the most significant events in the history of America. The presidential inauguration ceremony was held on January 20, 2021. It is an example of an important event with a great degree of conversation on Twitter. With this case study, we present our proposed approach (BUrST) in the real-world scenario. The sequential pattern of the event is to be followed, and it is essential to examine how the system reacts in different situations in terms of clusters that emerge, grow, and vanish. Numerous people participate in the event, and a sequence of event structure will be created. The US presidential election ceremony event's structure is stable, starting with a swearing in ceremony (12:00 pm ET) and ends with the celebration of the America Primetime impressive starts at 8:30 pm ET (90 min). Before the ceremony, all the associated entities are clustered from a Twitter text stream, and we have observed that after the end of the ceremony, people continue to talk about topics that were very popular during the event (united America, president for all Americans, COVID-19, history and hope, democracy prevailed). From these all observations, it is found that the proposed system can capture event-related topics and performing well to capture such fast evolving topics. The major highlight includes Swearing-InCeremony (12 pm ET), pass in review, presidential escort (after 3 pm ET), virtual parade across America, Fields of flags, Celebrating America Primetime special (8:30 pm ET).However, the most interesting period is during the ceremony itself. The proposed approach can capture fast evolving topics and create different clusters for them. The analysis for event burst is also interesting to study in this case study. As the proposed system captures the domain information, the analysis of burst events is also demonstrated, highlighting the import feature related to the event during real-time conversation and approaching threshold value. Users, who were watching the event online, actively participated in live streams, making this task easy for the proposed (BUrST) approach to track the burst nature of events in real-time scenarios.

10 Conclusion and future scope

Event detection and extraction in social media text stream is difficult even to deal with the trend from the extracted event pattern is a quite challenging and complex process. The textual information produced by multiple users at different times and locations plays a key role. Sometimes, user re-tweets on existing tweets, and it becomes bursty on social media, and it is also an important aspect of the modern social world. This research work's main motive is to develop an approach that combines all the related tweets online in a regular fashion and group them into a cluster based on their similarity index. Various features associated with online text streams are introduced, and a meaningful pattern of information is extracted from them, making the classification task easy. For checking the performance of the proposed approach at different stages, we compared it with the state-of-the-art methods addressing various concerns, and we found that the proposed approach is working very well. As from results, we have seen that the proposed approach is performing well as compare to other existing methods, including where BUrST is giving 0.69 (FA CUP), 0.68 (US Elections), and 0.72 (Enron mail stream) in event topic recall.

Similarly, in event keyword recall, the proposed approach performs better by giving 0.68, 0.59, and 0.76 in the FA CUP, US election and Enron mail stream. For event keyword precision, 0.61 is obtained for FA CUP, 0.58 is obtained in IS, and 0.72 obtained for the Enron mail stream compared to other existing approaches. On the other hand, numerous dimensions can be explored to extend it in the future. Contextual information of keywords for finding the bursty events in the text stream can also be included for the event detection field better to understand the role of context in event detection methods. Early detection mechanism for burst events can also help analyze the burst behavior patterns that are still not explored and can be studied to predict the various real-time events domain.

References

1. Fedoryszak M, Frederick B, Rajaram V, Zhong C, (2019) Real-time event detection on social data streams. In: KDD '19, August 4–8, Anchorage, AK, USA
2. Comito C, Forestiero A, Pizzuti C (2019) Bursty event detection in twitter streams. ACM Trans Knowl Discov Data 13(4):1–28
3. Feng W, Zhang C, Zhang W, Han J, Wang J, Aggarwal C, and Huang J, (2015) Streamcube: hierarchical spatio-temporal hashtag clustering for event exploration over the twitter stream. In: ICDE
4. Imran Muhammad, Castillo Carlos, Diaz Fernando, Vieweg Sarah (2015) Processing social media messages in massemergency: a survey. ACM Comput Surv 47(4):38
5. Allan J, Carbonell J, Doddington G, Yamron J, and Yang Y, (1998) Topic Detection and Tracking Pilot Study Final Report. In: Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop
6. Filatova E, Hatzivassiloglou V, McKeown K, (2006) Automatic creation of domain templates. In: Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, Sydney (pp 207–214)

7. Zhou X, Chen L (2014) Event detection over twitter social media streams. VLDB J 23(3):381–400
8. Aggarwal CC and Subbian K, (2012) Event detection in social streams. In: Proceedings of the 2012 SIAM International Conference on Data Mining (pp 624–635)
9. Li C, Sun A, and Datta A, (2012) Twevent: segment-based event detection from tweets. In: Proceedingd of the 21st ACM International Conference on Information and Knowledge Management CIKM (pp 155–164)
10. Xing C, Wang Y, Liu J, Huang Y, and Ma W-Y (2016) Hashtag-based sub-event discovery using mutually generative lda in twitter. In: Proceedings of the AAAI Conference on Artificial Intelligence
11. Cadenas JM, Garrido MC, Martínez R (2013) NIP - an imperfection processor to data mining datasets. Int J Comput Intell Syst 6(sup1):3–17
12. Lee P, Lakshmanan LV, and Milios EE (2014) Incremental cluster evolution tracking from highly dynamic network data. In Data Engineering (ICDE), 30th International Conference on IEEE (pp 3–14)
13. Fu Z, Sun X, Shu J, Zhou L (2014) Plain text zero knowledge watermarking detection based on asymmetric encryption. Adv Sci Technol 48:126–134
14. Becker H, Naaman M, and Gravano L (2011) Beyond trending topics: real-world event identification on twitter. In: Proceedings of the International AAAI Conference on Web and Social Media, Icwsm (pp 438–441)
15. Singh T, Kumari M (2016) Role of text pre-processing in twitter sentiment analysis. Procedia Comput Scis 89:549–554
16. Singh T, Kumari M, Pal TL, Chauhan A (2017) Current trends in text mining for social media. Int J Grid Distrib Comput 10(6):11–28
17. Carbonell JG, Yang Y, Lafferty J, Brown R, Pierce T, and Liu X, (1999) CMU Approach to TDT-2: Segmentation, Detection, and Tracking. In: Proceedings of the 1999 DARPA Broadcast News Conference
18. Orr JW, Tadepalli P, and Fern X (2018). Event detection with neural networks: a rigorous empirical evaluation. arXiv preprint
19. McMinn AJ and Jose JM (2015). Real-time entity-based event detection for twitter. In: International Conference of the Cross-Language Evaluation Forum for European Languages. Springer, (PP 65–77)
20. Guille A, Favre C (2015) Event detection, tracking, and visualization in twitter: a mention-anomaly-based approach. Soc Netw Anal Min 5(1):18
21. Hasan M, Orgun MA, Schwitter Rolf (2017) A survey on realtimeevent detection from the twitter data stream. J Inf Sci. https://doi.org/10.1177/0165551517698564
22. Atefeh F, Khreich W (2015) A survey of techniques for event detection in twitter. Comput Intell 31(1):132–164
23. Aggarwal CC, Wang J (2007) Data streams: models and algorithms data streams. Kluwer Academic Publishers, Boston, Dordrecht, London
24. Allan J, Papka R, and Lavrenko V, (1998) On-line New Event Detection and Tracking. In: SIGIR‘98, Melbourne, Australia, 1998 ACM, (pp 37–48)
25. Aggarwal CC and Subbian K, (2012) Event detection in social streams. In: Proceeding 2012 SIAM International Conference Data Mining, (pp 624–635)
26. Dong X, Mavroeidis D, Calabrese F, Frossard P (2015) Multiscale event detection in social media. Data Min Knowl Discov 29(5):1374–1405
27. Becker H and Gravano L, (2010) Learning similarity metrics for event identification in social media categories and subject descriptors. In: WSDM‘10, February 4–6, 2010, New York City, New York, USA
28. Mathioudakis M and Koudas N (2010) Twittermonitor: trend detection over the twitter stream. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of data ACM, (pp 1155–1158)
29. Osborne M, Moran S, McCreadie R, Von Lunen A, Sykora MD, Cano E, Ireson N, Macdonald C, Ounis I, He Y, et al (2014) Real-time detection, tracking, and monitoring of automatically discoveredevents in social media
30. Petrović S, Osborne M, and Lavrenko V (2010) Streaming first story detection with application to twitter. In Human language technologies: the 2010 annual conference of the North American chapter of the association for computational linguistics. Association for Computational Linguistics, (pp 181–189)

31. Hasan M, Orgun MA, Schwitter R (2016) TwitterNews: realtime event detection from the Twitter data stream. Peer J PrePrints 4:e2297v1
32. Paul D, Li F, Teja MK, Yu X, and Frost R (2017) Compass: spatio temporal sentiment analysis of US election what twitter says. In: KDD.ACM (pp 1585–1594)
33. Fung GPC, Yu JX, Yu PS, and Lu H (2005) Parameter free bursty events detection in text streams. In: Proceedings of the 31st International Conference on Very Large Data Bases (VLDB'05) (pp 181–192)
34. Qiaozhu M and Zhai CX (2005) Discovering evolutionary theme patterns from text: an exploration of temporaltext mining. In: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD'05). ACM, New York, NY (pp 198–207)
35. Sakaki T, Okazaki M, Matsuo Y (2013) Tweet analysis for real-time event detection and earthquake reporting system development. IEEE Trans Knowl Data Eng 25(4):919–931
36. Thorsten J (1998) Text categorization with support vector machines: Learning with many relevant features. In: Proceedings of the 10th European Conference on Machine Learning (ECML'98). (pp 137–142)
37. Stilo G, Velardi P (2016) Efficient temporal mining of micro-blog texts and its application to event discovery. Data Min Knowl Discov 30(2):372–402
38. Xie W, Zhu F, Jiang J, Lim EP, Wang K (2016) TopicSketch: real-time bursty topic detection from Twitter. IEEE Trans Knowl Data Eng 28(8):2216–2229
39. Yin Jie, Lampert Andrew, Cameron Mark A, Robinson Bella, Power Robert (2012) Using social media to enhance emergency situation awareness. IEEE Intell Syst 27(6):52–59
40. He Q, Chang K, and Lim E, (2007) Analyzing feature trajectories for event detection. In: SIGIR'07, July 23–27, 2007, Amsterdam, The Netherlands. Copyright 2007 ACM, (pp 207–214)
41. Kleinberg J, (2002) Bursty and hierarchical structure in streams. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (pp 1–25)
42. Wang X, Zhai C, Hu X, and Sproat R, (2007) Mining correlated bursty topic patterns from coordinated text streams. In: KDD'07, August 12–15, 2007, San Jose, California, USA
43. Kaliyar RK, Goswami A, Narang P (2021) DeepFakE: improving fake news detection using tensor decomposition-based deep neural network. J Supercomput 77:1015–1037
44. Ahmad I, Ahmed G, Shah SAA et al (2020) A decade of big data literature: analysis of trends in light of bibliometrics. J Supercomput 76:3555–3571
45. Venkatraman S, Surendiran B, Arun Raj Kumar P (2020) Spam e-mail classification for the Internet of Tfhings environment using semantic similarity approach. J Supercomput 76:756–776
46. Lee H, Lee N, Seo H et al (2020) Developing a supervised learning-based social media business sentiment index. J Supercomput 76:3882–3897
47. Daniel Jurafsky, James H. Martin, Parsing D, (2018) Dependency Parsing. Speech and Language Processing, (pp 1–27)
48. Hamdan H, Bellot P, and Bechet F, (2015) Lsislif : feature extraction and label weighting for sentiment analysis in twitter. In: SemEval, (pp 568–573)
49. Knuth DE (1997) The art of computer programming: seminumerical algorithms, 3rd edn. Addison-Wesley Longman Publishing Co. Inc, Boston, MA
50. Kenter T, and de Rijke M, (2015) Short text similarity with word embeddings categories and subject descriptors. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM 2015) (pp. 1411–1420)
51. Tang Q, Jian Q, Meng M, (2015) PTE : predictive text embedding through large-scale heterogeneous text networks categories and subject descriptors. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 1165–1174)
52. Aiello Luca Maria, Petkos Georgios, Martin Carlos, Corney David, Papadopoulos Symeon, Skraba Ryan, Goker Ayse, Kompatsiaris Ioannis, Jaimes Alejandro (2013) Sensing trending topics in Twitter. IEEE Trans Multimed 15(6):1268–1282
53. Aggarwal CC and Subbian K, (2012) Event detection in social streams. In: Proceedings of the 2012 SIAM International Conference on Data Mining (pp 624–635)
54. Blei DM, Ng AY, Jordan MI (2003) Latent dirichlet allocation. J Mach Learn Res 3:993–1022
55. Petrovic S, Osborne M, and Lavrenko V, (2010) Streaming first story ´ detection with application to twitter. In: HLT: Annual Conference of the North American Chapter of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics, (pp 181–189)

56. Xu X, Yuruk N, Feng Z, Schweiger TAJ (2007) SCAN: a structural clustering algorithm for networks. In: KDD: 13th ACM International Conference on Knowledge Discovery and Data Mining. New York, NY, USA: ACM, (pp 824–833)
57. O'Connor B, Krieger M, Ahn D (2010) TweetMotif: exploratory search and topic summarization for twitter. In: ICWSM, WW Cohen, S Gosling, WW Cohen, and S Gosling, (Eds). The AAAI Press
58. E Winarko, R Pulungan (2019) Trending topics detection of Indonesian tweets using BN-grams and Doc-p. J King Saud Univ Comput Inf Sci 31:266–274