Cluster Computing (2022) 25:3799–3817
https://doi.org/10.1007/s10586-022-03610-6

Real-time event detection and classification in social text steam using embedding

Tajinder Singh[1] · Madhu Kumari[2] · Daya Sagar Gupta[3]

Received: 26 March 2021 / Revised: 31 January 2022 / Accepted: 20 April 2022 / Published online: 13 May 2022

Abstract
Taming data will always be a significant challenge in online social networks. These networks are rapidly becoming the emerging source for users to explore the primary sources to seek information in the form of events. Rich informational data can be extracted from various social platforms like twitter text streams for direct insights into enduring topics and classifying them based on their similarities. To address the research issues of event detection and classification, we model events as evolving clusters over a period of time. The inability of conventional clustering algorithms to process the data streams mandates the use of a fast yet robust method. Therefore this work employs quick comparisons of data coming from social streams relying on a twin network known as the Siamese network, which can detect the novel event based on clustering by comparing their content dependent feature. We also trained dataset derived from the social text stream from twitter and other sources, where embedding encode every word representation mapped to a vector. This representation of word into real valued vectors provides a specific processing task for event classification. Finally, we compared the proposed technique with the existing methods, and the results obtained through several experiments are a clear indicator of the efficacy of the proposed scheme.

Keywords Event detection · Clustering · Embedding · Text stream · Social event

1 Introduction

Online social networks and social media platforms bring a new world of information explosion, which leads to generating new content related to various social events. In social networks,a text arrives in a massive stream of text segments. These text segments have meaningful patterns of information. In the modern world, maximum journalists use social media on priority due to its ease of access. Social media's flexible role facilitates the current researcher or information seekers to extract exciting and unique patterns that are otherwise difficult to discover [1]. Nowadays, in every social media, numerous kind of events occurs. Users post their sentiments and enhance its diffusion rate. Thus, an event E is defined as something that happens at some specific time T and location l. Examining the extracted text can be an elementary source for discovering and analyzing the events [2]. In this way, we can analyze users' behavior U by seeking their interest at different spans of time. Such type of analysis helps to make strategic decisions for an organization to facilitate its users. Event detection and classification tasks are closely correlated to the topic detection and tracking and play a significant duty in text mining [3]. Many researchers use this relation to predict the trend of an event. Usually,in the social text stream S, a stream is represented as $\{s_1, s_2, \cdots s_n\}$ at a time T where $T = \{t_1, t_2, \cdots t_n\}$. It facilitates social media users to

✉ Daya Sagar Gupta
dayasagar.ism@gmail.com

Tajinder Singh
nith2k14@gmail.com

Madhu Kumari
madhu.jaglan@gmail.com

1 Department of Computer Science and Engineering, Sant Longowal Institute of Engineering and Technology Longowal, Sangrur, Punjab 148106, India

2 University Centre of Research and Development, Chandigarh University, Punjab, India

3 Department of Computer Science and Engineering, Rajiv Gandhi Institute of Petroleum Technology Jais, Amethi, Uttar Pradesh 229304, India

recognize, focus interest, and take effects from imperative events in the early stages while still developing [4, 5]. For event detection, an online clustering mechanism is extensively utilized in several variations [6, 7]. The clustering mechanism is playing a crucial role in text streams to differentiate event-related text from non-event related text [8–10]. Clustering documents is a complex task which includes multiple users U where $U = \{u_1, u_2, \cdots u_n\}$. Numerous users can post a tweet τ_w on Twitter at a particular time t_α. The extraction process of tweets from twitter will be continuous, which is described as an event start time $t_{E_{start}}$ and event end time $t_{E_{end}}$. For a specific event e_α belongs to a series of events $E = \{e_1, e_2, \cdots e_n\}$, a trend can be predicted by analyzing these event's behavior. This scenario also helps to know the event's impact and its diffusion rate in the social culture, which is usually measured in terms of peak time of an event $T_{e_{peak}}$.

Along-with information and knowledge, social text streams also include unwanted text that needs to be cleaned before this data can be utilized for the tasks such as event identification and detection [11, 12], trend prediction [13], sentiment analysis, topic tracking, recommendation system, tagging [14] and many more. Usually, in social text streams, novel and evolutionary events have a significant contribution. If an event E is found to be novel in the text stream, a new cluster C_{new} is created for this event; otherwise, it will be a sub-event of an already available event(cluster), which contributes to the evolution of a particular event and helps in trend prediction. A set of events E in a text stream, $E = \{e_1, e_2, \cdots e_n\}$, one event can push others to occur with some combined probability of occurrence $Pr(E)$ represented as: $Pr(e_1) + Pr(e_2) + \cdots Pr(e_n)$. We considered a random variable R_i for the event e_i which maps a word to its belongingness with e_i. R_i can be defined for a word w which is present in a stream and an event e_i as in Eq. (1):

$$R_i(w) = \begin{cases} 1 & if\ w \in e_i \\ 0 & if\ w \notin e_i \end{cases} \tag{1}$$

The total number of events that will occur is the sum of all random variables $(R_1 + R_2 + \cdots + R_n)$. In social text stream, a particular event will occur in a specific time period. Therefore, expected value is defined as a average outcome of a social text stream of random events with similar meaning of lexicons which occurs in a particular time for longtime. Thus, the expected value of probability is computed by multiplying each of the possible outcomes by the likelihood each outcome will occur and then summing all of those values as defined in Eq.(2). Using linearity of expectation sum of events can be computed as:

$$EX_{e_i}(T) = \sum_{i=1}^{n} Pr(e_i) \text{ where } EX \text{ denotes expectation} \tag{2}$$

Eq.(2) is a probability of the occurrence of expected events from the text stream.In this research work, we are using Twitter text, which is extracted from Twitter API. Generally,the stream S is represented as $\{S = s_1, s_2, \cdots s_n\}$ which contains tweets Γ_w. We extracted data in a closed interval of time $[t_1, t_2]$, including a sequence of text streams in the form of tweets represented as: $\Gamma_w = \{\tau_{w_1}, \tau_{w_1}, \cdots, \tau_{w_n}\}$. As the steam sequence S is a combination of all the streams $\{s_1, s_2, \cdots, s_i, \cdots, s_n\}$, there is a stream containing a tweet τ_{w_i}. This tweet may or may not be linked to other tweets and streams; such tweets should be scraped or ignored from S.

This research work is significant for business analysis and strategic planning that deals with influences and disruptions that may occur because of the forthcoming events,and it may also help in detecting factors that influence the spread of a particular event. Compared to previously published work on event detection, this paper has some important contributions as:

a) Tracking Event's Impact: In social text streams, events can be tracked and classified mostly based on location. We have analyzed the impact of an event by finding its network influences at a particular time. In this research work, we classify the events first based on location, and thereafter we detected those events that have the highest impact at a particular time.
b) Online Clustering With Contexts:We have used online clustering for faster analysis of contexts in streams to put similar events together.Our model uses embedding with contextual information from text, network, and users; this gives our algorithm an edge for fast retrieval of the related pattern from databases and data streams, as shown in [15]. This work's core objective is to separate almost infinite unlabeled text into a finite and discrete set of distinguished events.
c) Novel Event Detection: An event is referred to as a novel event if there exists no cluster related to the upcoming events with substantial textual and busty support from data streams. It requires a careful and deep examination of text streams. In the proposed scheme,algorithms used not only the textual and topical similarities but also the contexts, which nullify the possibility of finding redundant events. A new cluster used to store this novel event and the upcoming related events.

d) Sensitivity Towards False Positives:Our event detection algorithm strongly discourages redundant and false events(rumors) due to robust filtering criteria and contextual linkages between text streams. It makes the proposed scheme sensitive towards rumors also.

e) The proposed approach is given in this research work to clean the collected text to extract the required data for analysis. Special symbols, abbreviation, short term used in text stream, emoticons are removed so that exact meaning of text can be extracted.

2 Literature survey

Various definitions of event detection and classification are available based on multiple descriptions of events and contextual information in the Twitter social text stream. Several surveys describing state-of-the-art event detection methods as explained in [16, 17]. Former approaches defined by various authors are classified into different ways to detect and classify from which most of the clustering mechanisms are based on textual similarity. In [18], the context of news wire documents is considered in which authors defined event detection as recognizing “prompt words” and classify events into “refined types”.

Twitter has public nature due to which it includes vast challenges. Despite these confrontations, twitter is used as a primary source of data in various research tasks focused on social text stream. In the text stream, it is quite challenging to expect prompt words due to the unstructured nature of stream text, and it is quite complex to recognize the features of unstructured text. Text in social stream contains various symbols (@, #, etc.) [15], short forms (hlo, hru, luv, etc.), which make social text noisy and challenging [19, 20]. Multiple approaches claim to solve this problem of text pre-processing in which supervised and unsupervised techniques are available. Content-based methods using supervised and unsupervised approaches for event detection are presented in [21].These constructs are also capable of classifying and detecting events from the social text stream. Unsupervised techniques used in this context are usually based on a clustering mechanism whose main aim is to organize similar kinds of events using dynamic clustering [9]. Similarly, online clustering in event identification comes in the real picture and this is used for topic detection, event detection, classification, and trend prediction. In former research, many authors explained and discussed streams clustering in social media for event detection, classification, and tracking problem. In [22], the authors considered the issue of online clustering, whereas [9] explained the new event detection and tracking. Authors developed a novel approach that consists of a single pass clustering and online adaptive filtering to handle evolving events in the news. Similarity/dissimilarity among the tweets cannot be assessed without content analysis. Wavelet transform for multi-scale event detection in temporal and spatial scales is used in [23], and events are analyzed based on their temporal scale. Text similarity can also be measured using graphs in which graphical representation is introduced by authors to compute text similarity and compare it with clustering.

Usually, if we consider spatial and temporal information of events, contextual information associated with event changes over time, and people use numerous words to explain event topics. Moreover, the description of the topic and its probability distributions of the underlying topic representations also change. Such kind of scenario provides a deep sense of the event topic and helps to evaluate the real-time situation clearly and effectively, as described in [11]. Similarly, in [24],the authors addressed the problem of dynamic change of word vocabulary with change over time. Vector space models are usually used to represent tweets. A fixed vector dimension is used in advance due to the fixed size of word vocabulary because, in text streaming scenarios, word vocabulary dynamically changes over time. Recent years have seen a flow of adopting rapid numeric vector representation to extract and capture words’ semantic. In [25], authors provide detailed information of vector representation of text whereas, for statistical language modeling, early approaches developed counts number of words based on co-occurrence using n-gram. They generate semantic vector using the count of vectors. Recently, for word co-occurrence classification word2vec is used to train neural networks for large texts [26]. In this weights encode words in the input to a neural network for another purpose or task to represent words in a solid vector form. Many other approaches and models are available and designed since word2vec, and pre-trained semantics vectors are also available,making this task simple.

Many recent research studies [27, 28] suggested using an incremental clustering mechanism whose aim is to concentrate and to solve the problem of event evolution [29]. Designed mechanism and models are incrementally updated and when novel text appear on a text stream. Sometimes, such methods may not be accurate and feasible in a vast text stream due to the scale of updates. Therefore, in this research paper, we solve the problem by updating the weights layers of networks on the arrival of new data in the text stream. The score is computed after updating the weight on a hidden layer, and this mechanism provides better results. Our proposed approach deals with real-time data and achieves event detection, and classification as an online cluster mechanism is addresses the event progress over time. As part of this online text stream for event detection and classification, we extracted sets of events from a Twitter data stream. We analyzed the impact of scale onthe variation of the Twitter text stream. Our research is optimized for low complexity in terms of time-space complexity to attain high quality to explore, detect, and classify events in text streams. However, none of the works presents an efficient algorithmic approach for event detection, classification, and tracking in massive text streams in real or substantially less time. For this purpose, we developed a novel approach that uses a fast discriminative model (Sesame Network) to detect trendy events from the large corpus of text stream and identify its influence impact in future tweets.

3 Problem formulation and methodology

In this section, we are describing the problem formulation in event detection and classification. The designing mechanism is also introduced in this section, which provides the refine information to extract useful information related to events.

3.1 Problem definition

This section defines the problem statement in the domain of event detection and classification in social text streams. A significant issue in the social text stream includes the role of pre-processing in crawled social text. In our research work, firstly, we crawled text from the Twitter text stream. The extracted text is to be processed and which will act as an input for the perceived events. Next, we discussed the proposed Methodology with essential terminology. The main idea is to design a real-time system. We carried our experiment on the query-based crawling of text streams. Our designed interface can collect and analyze the streams generated and posted by numerous Twitter users. For analyzing the novel event from the text stream, clustering methodology is used. In this case, clustering will cluster all the similar events together, and if any new event arrived, then a new cluster is to be assigned based on its novelty. In the previous event detection and classification task, contextual information is missing due to which many keywords associated with an event can be ignored. Because word related to the event may occur in another form also, and upto date, this problem is not addressed. Therefore, our proposed approach is also including the contextual information of events in the form of text. For this purpose, a parse tree is generated, which will compare the event's word with the root word, and then a cluster will be assigned based on their similarity index. As a cluster represents an event in our model, the event's frequency is checked regularly, and this cluster's field value is updated. We developed a weighting mechanism, which will update the value of the event-related text. We have employed a siamese neural network as a discriminator because it is sensitive to the input change. It acts as an adaptive filter, and it can compute the updated weights of a cluster efficiently in different text streams. Through this, we achieved an effective and relatively compressed input of text stream and an effective filtering technique capable of handling the long term dependency of tweets in a social text stream.

In this research work, we assume that the designed approach can be appropriate for various real-world situations and provide a faster way for event detection. The developed approach is interesting and relevant for short text messages on twitter text streams and other short messages including emails.

3.2 Designing methodology

Text in the form of 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 [19]. After this process,the clustering mechanism is to be implemented as the data is continuously coming, and it's quite challenging to analyze its features to combine. Thus, to obtain and handle this challenge, we used an online clustering mechanism to build up information comparative to the text stream, which is already classified and normalized to assign cluster accordingly. To make this concept understandable, we have used a few definitions to provide a clear situational perspective.

Definition 1 (*Social Text Stream*) A text stream $S = \{s_1, s_2, \cdots, s_n\}$ is a set of all the streams collected during a closed interval of time $[t_1, t_2]$ where the initial time is t_α. The sequence of text stream is represented as $\Gamma_w = \{\tau_{w_1}, \tau_{w_1}, \cdots, \tau_{w_n}\}$, where $\tau_{w_1}, \tau_{w_1}, \cdots, \tau_{w_n}$ are the set of tweets extracted from the Twitter stream.

Definition 2 (*Social media topic and event*) Topics are the combination of multiple documents that are directly or indirectly related to an event or activities [30]. Usually, topics are linked with real-world events associated with some other kind of events in social media. Therefore, an event E is something that occurs at any time t, at any location l. Events occurring scenario will be in a continuous fashion for a period known as event start time $t_{E_{start}}$ and event end time $t_{E_{end}}$.

Definition 3 (Clustering) Clustering of text stream is necessary to categorize the similar contents in the summarized form [9]. For this purpose, text streams $S =$

$\{s_1, s_2, \cdots, s_n\}$ are distributed among clusters (C), which is defined as $C = \{c_1, c_2, \cdots, c_n\}$ where $c_1, c_2, \cdots, c_n \in S$, such that $S = \cup(c_1, c_2, \cdots, c_n)$. Each of the object s_i belongs to the cluster c_j where $c_j \subset C$.

Definition 4 (Cluster Creation and life span of a Cluster) As the text streams are distributed among clusters C which is defined as $C = \{c_1, c_2, \cdots, c_n\}$ where $c_1, c_2, \cdots, c_n \in S$, such that $S = \cup\{c_1, c_2, \cdots, c_n\}$. Each of the tweets from the text stream, such as s_j belongs to the cluster c_j based on similarity at a time t_j where $c_j \subset C$. Consider c_{lf} is the representing a life of a cluster, t_α is the creation time of cluster, t_χ is the time when the last event keyword is entered in a cluster, η is a survival time function for a cluster after creation (even no tweet added after creation time) of a cluster whereas γ is a user-defined; which indicates the dynamic nature of the cluster. The whole scenario is represented as the Eq. (3) as follows.

$$\eta + 2^\gamma \times (t_\chi - t_\alpha) \tag{3}$$

Definition 5 (Event novelty detection) Novelty concept of an event $e_{\beta_{novel}}$ in text stream is associated with the creation of a new cluster C_{new_t} is represented as:

$$e_{\beta_{novel}} \propto C_{new_t} \tag{4}$$

Therefore, in Eq. (4), a dynamic nature of text stream which changes with time can be understand and clustered as required based on upcoming tweets. New cluster will be assigned when a novel keyword will exist from social text streams which help to analyze novel event from social text stream.

Definition 6 (*Similarity measurement*) Various clusters are created during event detection in the text stream. Therefore, on the basis of the similarity function cluster is assigned to a particular event. Thus, the similarity function $sim_f(s_f)$ holds the information about the content changed within the clusters C and changes in the stream of text in a closed time interval $[t_1, t_2]$. The upcoming social text stream is assigned to a cluster based on similarity index where corresponding values of the forthcoming events are determined in which the mean (λ) and standard deviation (σ) of all closely match event objects are maintained.

Definition 7 (Threshold value) The threshold value of a cluster is used to generate a new cluster for a similar kind of events when they occur in huge amount. This value is directly associated with the similarity measurement (definition 6) in which mean, and standard deviations are directly involved. Therefore, for this purpose, we set the threshold value $(\mu - 2 \cdot \sigma)$.

Considering an example in which this threshold mechanism is applied to various feature tweet words τ_{w_n} in the social text stream s_n. If the upcoming event in the social text stream is closely matched with tweet word τ_{w_n} multiple times, and its value exceeds $(\mu - 2 \cdot \sigma)$ then, a new cluster will be created, which will represent the event words similar to τ_{w_n}.

3.3 Text stream pre-processing

To remove unwanted text from social media pre-processing is essential to reduce noise; verbose words and slangs are needed to convert into genuine words. Generally in normalization stop list [7], tagging (# tags & @ tags), URL, are removed to discover meaningful pattern from the collected text stream. As we are using the Methodology proposed in [19] for text pre-processing, which is effective to tame text in terms of noise, which includes emoticons, misspelled or incorrect words, folksonomies, and slangs, etc. This normalization approach is an endeavor to observe the outcome of pre-processing on the twitter text stream for the fortification of event detection and classification, mainly in terms of actual event word [19, 31].

Algorithm 1: Text Stream pre-processing

```
Initialization;
Input: A continuous stream of tweets $\Gamma_w = \{\tau_{w_1}, \tau_{w_1}, \cdots, \tau_{w_n}\}$, unidentified word
    $w_{ui}$, Cluster survival time $\eta$, $t_c$ is a creation time of cluster, $t_\lambda$ time when
    cluster will not accept more events, i.e., the time when the last event
    entered, $t_\eta$ is the time span, and $C_{life}$ is the life or survival life of a cluster.
Output: Active cluster $C$ with a collection of classified words $w_i$ in terms of events
    from text streams $S$.
begin
    if $\tau_i \rightarrow w_i$ /*Tweet is identified assign cluster $c_i$ on the basis of $sim_f(s_f)$*/
     then
        $c_i = c_i + 1$; /*Update the cluster value by 1*/
        $\tau_w \leftarrow \tau_w(url)|@|splsymbol|\#\ tag$;
        $\Gamma_w = \{\tau_{w_1}, \tau_{w_1}, \cdots, \tau_{w_n}\} \in S$, Collection of refined pre-processed Tweets,
          which belongs to the Twitter text stream $S$.
        while not the end of stream do
            Compute indicator dataset of text stream corresponding to $i$.
            $i \rightarrow i + 1$; /*Obtain the next tweet from the stream $t_{w_i}$*/
            Apply the above If condition for the collection of refined pre-processed
              Tweets, which belongs to the Twitter text stream $S$ for the new
              coming tweets
        end
        for each cluster $C = \{c_1, c_2, \cdots, c_n\}$ do
            Similarity of each upcoming tweet is computed to assign cluster on the
              basis of the similarity index $sim_f(s_f)$
        end
    end
    else
        /*a new cluster is to be created as the new event-related word is
          detected.*/
        $C_{new} \leftarrow C$;
        $C_{new} \leftarrow CreateCluster(C_\alpha)$ where $C = C \cup C_\alpha$ ;
    end
end
```

4 Methodology

The proposed Methodology is an efficient incremental 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. Assigning weight to the upcoming text stream is the main motive of the proposed methodology. All the collected text streams are pre-processed [19] using steps described in Algorithm 1. Assume that T_w denotes the collection of whole pre-processed text stream where $T_w = \{t_{w_1}, t_{w_2}, \cdots, t_{w_n}\}$ which can be represented by a $m-$dimensional vector. Let x be an ordered collection of m event-related tweets $\{\tau_{w_1}, \tau_{w_2}, \cdots, \tau_{w_n}\}$, and is written as $x = \{\tau_{w_1}, \tau_{w_2}, \cdots, \tau_{w_n}\}$. We have a stream of text having n points $T_w = \{t_{w_i}\}$ where $i = 1, 2, \cdots, n$ and $t_{w_i} \in \mathcal{R}^d$ in which the function of pairs is associated with two categories of labeling text information $T_{w_{sim}}$ and $T_{w_{nsim}}$. If $(t_{w_i}, t_{w_j}) \in T_{w_{sim}}$ are denoted having similar features when (t_{w_i}, t_{w_j}) are matching with their feature similarity and are neighbor pair (N) also in a metric space and belong to the same class labels otherwise $(t_{w_i}, t_{w_j}) \in T_{w_{nsim}}$ where (t_{w_i}, t_{w_j}) are far away (K) and not sharing standard class similarity features.

The main motive behind using *embedding* is to map data to precise space to obtain reliable representation [32, 33]. Suppose we want to create Embedding leading to a M-bit hamming embedding of $Z \in \mathcal{R}^{G\times n}$ and $X \in K^{M\times n}$ [34], which is obtained using the word2vec model with a skip-gram model. Thus given a vector $w_m \in \mathcal{R}^G$, the embedding depends on the input layer and the contextual information associated with the event's word at the output layer [35]. Figure 1 is depicting the simplified version of the text stream, S where H is the size of the hidden layer. $S \times H$ is representing the weights at the input and output layer as well of the matrix M. Each row of M is an n-dimensional vector representing s_w the associated words in the input layer.

In skip-gram, the input vector is representing as v_{w_i}, and the hidden layer input H is defined as in Eq. (5):

$$H = M^T x = M^T(k) = S_{w_i}^T \tag{5}$$

The whole scenario is calculated, and the hidden layer passes the weighted sum of input to the next layer, i.e.,the output layer. Therefore, a new weight matrix $W' = \{w'_{ij}\}$ is

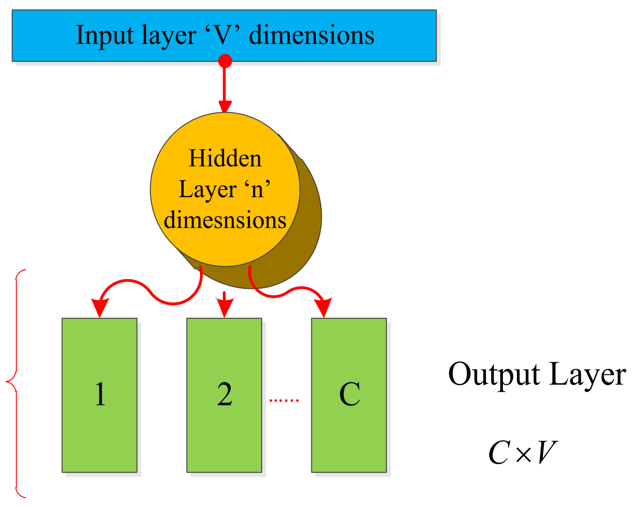

Fig. 1 Simplified representation of text streams

present, which is represented as $H \times S$. Thus, the weight helps to compute the score $score(s_j)$ for each word in the text stream S. The normalized exponential function is used to categorize the probability distribution of weights in a text stream.

$$\sigma(z)_j = \frac{e^{zj}}{\sum_{k=1}^{K} e^{zk}} \tag{6}$$

After the distribution of weights,generally fast online clustering is employed [8, 10]. The clustering mechanism helps to categorize the events on the basis of their similarity and weight as well. The output layer is updated,and for this purpose, Eq.(6) is used. For learning updated weight, each embedding vector for every word is multiplied with weight. Consider a stream S whose random pair is represented by P. P^{α} and P^{β} are its pair text whose vector representations are P^{α}_j and P^{β}_j respectively, where all words having the same dimension v. $n(P^{\alpha})$ and $n(P^{\beta})$ are its length associated with a vector, which indicates the number of words in the text P^{α}. Assume that the length of the text P^{α} and P^{β} is the same, then $n(P^{\alpha}) = n(P^{\beta}) \forall P \in S$. $w_j, j = \{1, 2, \cdots, n\}$ are the weights which are needed to be learned in a stream of text and represented as Eq. (7).

$$\forall l \in \{\alpha, \beta\} : n(P^l) = \frac{1}{n} \sum_{j=1}^{n} w_j P^l_j \tag{7}$$

5 Online fast clustering approach (OFCA)

In this section, we are going to describe the online fast clustering mechanism which is used for event detection and classification. Various steps used for analyzing events behavior are presented in Algorithm 2.

Algorithm 2: Procedure for event detection and classification in various text streams using fast clustering

```
Initialization;
Input: A continuous stream of tweets $\Gamma_w = \{\tau_{w_1}, \tau_{w_1}, \cdots, \tau_{w_n}\}_{\tau_w=1}^{\tau_w=n}$, Cluster
       survival time $\eta$, $t_c$ is a creation time of cluster, $t_\lambda$ time when cluster will
       not accept more events, i.e., the time when the last event entered, $t_\eta$ is the
       time span, and $C_{life}$ is the life or survival life of a cluster. The weight is
       associated with text $\{w_j^{(\Gamma_w)}\}_{\tau_w=1}^{\tau_w=n}$ and cluster number $K$.
Output: Event detection and classified w.r.t. creation of new cluster and analyze
        the active cluster $C$ with a collection of classified words $w_i$ in terms of
        events from textstreams.
begin
    for $\tau_w = 1$ to $\tau_{w_n}$ from text stream
     $\{s_1, s_2, \cdots, s_n\} \rightarrow \{\tau_{w_1}, \tau_{w_1}, \cdots, \tau_{w_n}\}_{\tau_w=1}^{\tau_w=n}$ do
        Compute indicator dataset of text stream corresponding to $W^{\Gamma_w}$ and
         constitute the indicator subset $C_{\tau_w}^{(0)}$ that indicates $\Gamma_{w_0}$.
    end
    Initialize the $K$ initial clusters' centers $\{\tau_{w_j}^{\tau_{w_n}}\}_{j=1}^{K}$ of $\{C_{\tau_w}^{(0)}\}_{\tau_u=1}^{\tau_{w_n}}$ and set the
     cluster label.
    for $\tau_w = 1$ to $\tau_{w_n}$ do
        Divide $\{C_{\tau_w}^{(0)}\}_{\tau_u=1}^{\tau_{w_n}}$ into $K$ clusters by using the above "for" loop.
    end
    Re-compute the new cluster centers $\{\hat{\tau_j}^{\tau_{w_n}}\}_{k=1}^{K}$ of $\{C_{\tau_w}^{(0)}\}$
    Analyze the new cluster head and recompute the new cluster centers as:
     $\{\tau_j^{\tau_{w_n}} = \hat{\tau_j}^{\tau_{w_n}}\}_{k=1}^{K}$
end
```

5.1 Procedure

The proposed methodology 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 that represents an event. Further, it recognizes the events based on their event keyword, and the contextual information is also considered. If any new event arrives in the text stream, tweets $\Gamma_w = \{\tau_{w_1}, \tau_{w_1}, \cdots, \tau_{w_n}\}$ at the same time cluster c_i will be generated. In the very first step where any tweet enters, then a new cluster will be generated. An indicator is established and computed for text stream corresponding to W^{Γ_w} and constitute the indicator subset $C^{(0)}_{\tau_w}$ that indicates Γ_{w_0}.

The optimal value for the number of clusters K is determined using a gap statistic approach. The gap statistic method can work on any type of clustering approach. In this approach, the maximum within intra-cluster distinction for diverse values of K their predicted values under void reference distribution of collected text stream. In assigning optimal value, first of all, this will cluster the observed text in different clusters from $K = 1, 2, \cdots, K_{max}$ and will compute the matching values with feature value within the intra-cluster. After that, it will create another set of values with a random uniform distribution such as cluster $K = 1, 2, \cdots, K_{max}$ and calculate the estimated difference information as a deviation of the observed event value $W_{k_{observed}}$ from its expected feature value $W_{k_{expected}}$. The difference between these two values is computed. To give an optimal weight, a cluster with the smallest value of K is selected such that the difference between the statistics can be minimized. Furthermore, the cluster center is to be initialized, and the value of the cluster label is to set. Analysis of the cluster is to be evaluated; the new cluster head is to be recomputed. With this new value of the cluster head, the cluster center is also recomputed and, its value is to be processed further for taking any decision regarding events identification. Therefore, an updating function is required, which will update the value of cluster parameters on discovering a new text stream after assigning a cluster to a particular event. For this purpose, we have used the gradient descent method with the average learning rate η for updating weights. In this work, we are also describing the use of the siamese network for measuring text similarity.

5.1.1 Siamese neural network and its architecture

These are a special type of neural network architecture used to differentiate the two inputs. The siamese network helps to learn the input similarity, and it contains two or more similar networks (similar configuration, parameters, and weights). These networks are widely used in measuring similarity and a relationship between two text/images. Numerous applications are available that use the siamese network, but in our research work, we are using it to analyze the text-similarity in social text stream for events classification.

Therefore, using the siamese network,the vector representation of text is received at the output layer. Using the Gradient Descent method, with learning rate η is used for weight updating. From Eq.(2), weights are distributed where $z, j \in \{1, 2, \cdots, n\}$ which are needed to be updated to create a weighted sum of embedding where weights are only related to rank order based on probability. Distribution of weights (Eq. (6)) can be represented as [32]

$$\forall j \in \{1, 2, \cdots, n\} = z_j - \eta \frac{\partial}{\partial w_j} l(t(S^{\alpha})(S^{\beta}))$$

On applying skip-gram, this representation can be defined as

$$\begin{aligned}\forall j \in \{1, 2, \cdots, n\} =& z_{j_1} - \eta \frac{\partial}{\partial w_{j_1}} l(t(S^{\alpha_1})(S^{\beta_1}))\\ &+ z_{j_2} - \eta \frac{\partial}{\partial w_{j_2}} l(t(S^{\alpha_2})(S^{\beta_2}))\\ &+ \cdots + z_{j_n} - \eta \frac{\partial}{\partial w_{j_n}} l(t(S^{\alpha_n})(S^{\beta_n}))\\ &\Rightarrow Z_j - \eta \sum_{j=1}^{n} l(t(S^{\alpha_n})(S^{\beta_n}))\end{aligned}$$

Thus, new and old weight difference is computed as [32].

$$Z^{new}_{w_1} = Z^{old}_{w_1} - \eta(EI)_j H_i \tag{8}$$

From this new weight of vector at the output layer online fast clustering approach is to be implemented (Algorithm 1), which will cluster all the similar weight functions together with which it will be easy to identify the novel events very well [36, 37].

Therefore, we can say that the reason behind using Gradient descent is subject to extract event based on local minima. This approach is used in machine learning which helps to minimize loss and the impact of loss. Therefore, a L2 function is implemented to evaluate the performance of the proposed approach and it is observed that in this process we got a accurate predictions. Another main advantages of using gradient descent is that it has a power to handle the situation nicely towards the minimum value which can be covered by this function if it is complex in nature. Another reason to use this function is that we can fix the learning rate mechanism and it does not matter what will be the decay or weight loss during evaluation. Due to this reason of fixing weights, it does not degrade the performance. In text stream, since each event related semantic from text stream is used to analyze the weight of every lexicon in our Siamese neural network. The whole text stream batch is used to determine the each weight adjustment in this experiment and the various parameters are adjusted as our gradient decent is very comfortable by making weight adjustments after every data instance which adjusts weights based on a forward pass through the neural network, and the weights are updated immediately, after which a forward pass is made with the next event instance in the social text stream. This volatile nature of gradient decent process with greater fluctuations helps to make sure that a global cost function minima associated with text stream event is originated and has a better chance to analyze its impact on aspects. the parameters associated with these equations are adjusted on the basis of pre-trained feature embeddings which also helps to reduce the computing time. For establishment of various parameters fine tuning of the various event related keyword's from social text stream at different layers of embeddings are inherited using Distance Layer class of keras. After this parameters related to similarity between the various events related semantics is to be tamed by tf.GradientTaps. It will record and compute every parameter value which occur inside the network and will update the model weight at every step. In this way, the various parameters associated with embeddings are adjusted to analyze the internal behavior of the proposed model.

5.2 Procedure for event detection and classification

Various steps used for the event's weight distribution at the output layer is presented in Algorithm 3.

Algorithm 3: Procedure for the event's weight distribution at the output layer.

```
Initialization: Phase 1;
Input: A pre-processed form of Twitter Text stream
        $\Gamma_{w_p} = \{\tau_{w_{P_1}}, \tau_{w_{P_2}}, \cdots, \tau_{w_{P_n}}\}$ for the active Cluster $C$ from text stream $S$.
Output: Classified events and their combined version,which will calculate the
        frequency (weight) of a particularevent's related word with its contextual
        information as well.
begin
    if $N, K \to \{\tau_{w_i}, \tau_{w_i}\}$ then
        apply word2vec model with the skip-gram model as:
        $S \times H \to M$, distributed weights between the input layer and the output
          layer.
        $H = M^{T_w}x = M^{\tau_w}(k) = S_{w_i}^{T_w}$ where $S$ is a stream of text and is $H$ a
          hidden layer.
        $H \times S \to W' \to \{W'_{ij}\}$, weights of the hidden layer are passes to the
          output layer.
        $score(s_j) \to \forall\{\tau_{w_1}, \tau_{w_2}, \cdots, \tau_{w_n}\}$, For each event,a related word score is
          computed.
        $\sigma(w)_j = \frac{e^{zj}}{\sum_{k-1}^{k} z^{zk}}$, Weighs are distributed at the output layer.
    end
    Phase 2: Calculation of weight differences;
    Input: Weights at the output layer which are computed using $\sigma(z)_j$
    . do
        update weights at the output layer using generic weight updating equation
        $\forall l \in \{\alpha, \beta\} : n(P^l) = \frac{1}{n}\sum_{j=1}^{n} w_j P_j^l$
    while weights are updated apply online fast clustering (refer Algorithm 2);
    if weights are updated then
        calculate the difference between new and old weights.
        $Z_{w_1}^{new} = Z_{w_1}^{old} - \eta(EI)_j H_i$
    end
    else
        update weight and calculate weight difference $Z_{w_1}^{new}$
    end
end
```

From the above scenario, the weight of upcoming events has computed. It is also observed that when a similar event arrived, the value is updated based on the similarity, as described earlier. We filter out the various related events and compute weight at different layers to understand the cluster links. Thus, when weight is successfully updated, our approach notes down the new weight in the current time span. For any clusters that are not updated, we re-evaluate the text stream,and unique cluster-ID is to be assigned based on event arrival by linking clusters where appropriate, we update the weight. Furthermore, after weight computation, events classification is to be performed, and the error rate is to be computed given in the next sub-section. Algorithm 4 is providing the refine step by step information to classify events. For this purpose, we are using the L_2 function. The mechanism is described in the Algorithm 4 as given below:

Algorithm 4: Classification of event related keywords.

```
begin
    Apply $L_2$ function on $Z_{w_1}^{new}$ to compute the error in text streams.
    do $S = \sum_{i=o}^{n}(y_i - h(x_i))^2$;
    Calculate the total length of text as:
    $n_{max} \rightarrow compute$
    if weights are not updated then
        apply sub-sampling & linear interpolations
        $\forall l \in \{\alpha, \beta\} : n(P^l) = \frac{1}{n}\sum_{j=1}^{n} w_j P_j^l$
    end
    else
        repeate do
    end
    . Output: classified events
end
```

5.3 Error function

In this segment of the article, we intend the usage of the error function. The error function provides the motivation to analyze the performance of text, which helps to compute its actual score [38]. Assume that ε is the error function. Therefore, when clustering is implemented on the text stream S, some information context can be lost. The lost information can be significant or not, but necessary to compute the error rate to know the impact. Let c_v is the cluster from a set of the cluster C on which fast clustering is to be applied and the loss of the information is written as:

$$l^{\varepsilon} = \frac{1}{e_{w_i}} \sum_{c_v \in C} \varepsilon(c_v)$$

Where e_{w_i} is the number of identified event words from the stream from the cluster c_v.

Furthermore, consider (X, A) is a space, which is computable besides $S \subset \mathcal{R}$ be a closed subset. Thus, a function $E : X \times A \times \mathcal{R}$ is simply known as an error function if it is measurable. In our experiment, we are using the L_2 error function, which is also known as the least square error (LSE). This error function always bounces a stable solitary solution. This error function lessens the squared difference between estimated and existing target values. The representation is defined as:

$$S = \sum_{i=0}^{n}(y_i - h(x_i))^2 \tag{9}$$

The error between the estimated data and the target data value, which will help to make a decision about data values are to be computed where **squared** differences between the true value (y_i) and the predicted value (x_i) are calculated in Eq.(9). Due to its simplicity and easy to compute nature, we consider this error function. It also computes value with the adjustment of weights. Moreover, our proposed approach is also applicable to variable-length text streams. For this purpose, total n_{max} weight is required to compute. Therefore, to find weights for stream text with length $m \leq n_{max}$, we use sub-sampling and linear interpolation. This function's main purpose is to identify the loss of text where the length of the text is variable. Thus for random text T^r, find the sequence of real-valued indices as:

$$I(T^r, j), j \in \{1, 2, \cdots, n(T^r)\}$$

$$\forall j \in \{1, 2, \cdots, n(T^r)\} : I(T^r, j) = 1 + \frac{(j-1)(n_{max}-1)}{n(T^r)-1}$$

Furthermore, we will calculate the novel weights $z_j(T^r), j \in \{1, 2, \cdots, n(T^r)\}$ for the words in T^r using linear interpolation where $\in$ is randomly trivial.

6 Experimental study and data sets

The proposed Methodology is evaluated, and several experiments are done to establish effectiveness and efficiency. For this, we compared the proposed approach with two states of art approaches using three different data sets. In this section, we are discussing the existing approaches,such as CNN and VSM. We also explained the data sets used in this research, which are the Twitter text stream, Enron Mail stream, and IMBD text stream.

6.1 Convolution neural network (CNN)

Main motive behind CNN in the field of text analytic is to enable the machine to classify the data to extract knowledgeable patterns. The perceived information can be used further for making a decision that helps for analysis & classification, media recreation, recommendation systems, natural language processing, and many more.It consists of multiple hidden layers but contains two basic layers named as an input and an output layer. Hidden layers further consist of multiple convolution layers. In this way, the weights are distributed from the input layer to the output layer.

6.2 Vector space model (VSM)

VSM is also known as the similarity model, as it is based on the idea of similarity. In this model, it is assumed that the significance of a text document to query is almost the same as the document query similarity. BOW (Bag of Word) method is used for both,i.e., the text documents and the queries. VSM is very simple and easy to use in computing text similarity, and in this model, term weights are not binary. It helps to compute the similarity of documents and text in a continuous manner between queries and documents. Therefore, it is used widely in computing text similarity.

6.3 Evaluation mechanism and dataset

A brief description of text stream evaluation includes various steps is explained below:

– *Data Collection* Tweets are extracted from the twitter text stream, and for this purpose,Twitter API is used. We concentrated on the twitter text stream, IMBD text stream, and Enron mail stream in our research task. We collect the Tweets related to these data sets, and accordingly,the event's related keywords using hashtags are applied at different time slots of streaming text for extracting similar data. For extracting events text, we are also considering the contextual information. For this, a parse tree will be generated, which includes a root word, and on that related event's words, it will be classified from the extracted stream of text. The extraction process was divided into various time slots to collect an average amount of tweets according to the event's textual characters. With timestamp information in the text stream,the data set can be converted into streams [21]. Usually, sender and receiver information is attached in the email stream, which generated a complex network structure. Therefore, to understand the email network structure, it can be considered a tweet with a single sender and multiple receivers.
– *Mining the ground reality* Extracted text includes a huge number of hidden information. The concealed information includes information related to real facts. Therefore, to understand the reality of the text's awe-inspiring amount of influences can be entailed to extract features manually. In other words, we can say that we cannot rely totally on the extracted features from the stream or the features visible in data sets. For each topic, the ground reality contains multiple keywords on a particular time slot in which theme and features appear in the conventional reports.
– *Topic Detection*: As each extracted event should be based on the topic. The topic decides the relevancy of text in the upcoming stream, which is to be evaluated on each time slot. The extraction mechanism is divided into various time slots,and we merely consider the text from the stream as input with ground reality.
– *Evaluation of topic detection output with ground reality* Self-detection of the theme in a stream of text is evaluated using split error and JS Divergence. This is achieved by comparison on computed at the top *m* extracted topic in the stream.

6.4 Data sets

Twitter Text Stream:Twitter text stream tweets are crawled from the Twitter social network. Each social entity comprises the network structure and textual information in a tweet related to the query keyword. Some important features are considered as described in [23] where we crawled a total of 2356293 tweets from a stream in various time slots. Every text available in the text stream is related to different nodes. These nodes include the information of the sender and the receivers either in the way of direct indications from senders or their followers in the situation of transmission messages. On average, each stream object contained about 84 nodes per tweet, and each stream consists of 75-1300 tweets based on the popularity of the topic. After filtering these tweets in pre-processing, we have considered (250-550) different topics from the entire dataset; these topics may or may not be semantically distinguishable (Fig. 2).

Enron mail stream With the use of timestamp information in the Enron mail stream,we have transformed this data into text streams as described in [23]. In this stream, each object is associated with textual information. The textual information is associated with sender and receiver information, which generates a structure of the network. Thus, to seek and understand the structure of the email network, where mails are like tweets, can have multiple receivers are associated with a single sender. Usually, this dataset has normally one sender and one receiver. 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,we found that few emails were having duplicated entry and invalid text; therefore, these entries are weeded out from the final dataset. The Enron email data stream contained a total of 617,432 emails. We also eliminated the emails that did not have valid sender and receiver email identifiers, and finally, we got 459411 emails with valid information.

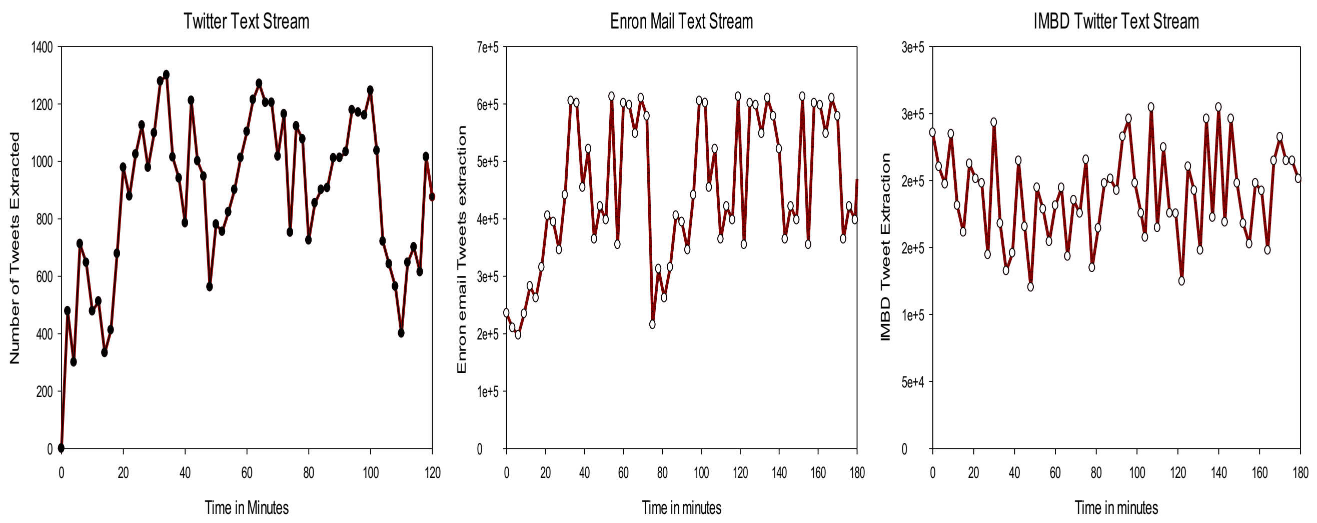

Fig. 2 Mechanism to extract tweets from twitter text stream: twitter text stream, enron mail stream and IMBD text stream

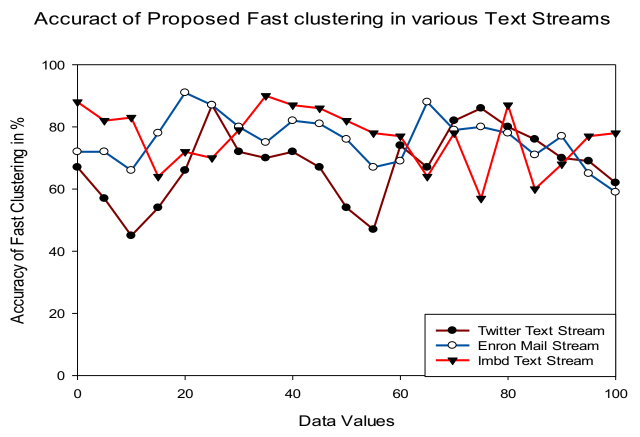

Fig. 3 Accuracy of the proposed fast clustering approach to detect events in various text streams

IMBD text stream In the IMBD data set again, we followed the time stamping mechanism to convert it into a stream of text [23]. This data set contains 24404096 ratings and 668953 tag applications across 40110 movies. For the streaming purpose, timestamp helps arrange the objects in the stream objects to maintain the chronological order. We weeded out the unwanted information from the collected stream in pre-processing such as fake or duplicate reviews, incomplete information, invalid user, folksonomies', and slangs. This data was created by 259137 users between January 09, 1995, and October 17, 2016. But after the normalization stream contained 218705 valid entries. Thus, from the above Fig. 3, it is very clear that the data collection mechanism started many days before finding useful patterns of events. The topics of the event are considered by various duration, which are divided into different time slots. We processed the extracted tweets for experiments, and only valid tweets with genuine information are considered, as explained in the data sets description. The proposed fast clustering approach is tested for all text streams, and it reflects the diverse behavior of the target events. As in the Twitter text stream, it was difficult to capture more information for the dedicated event due to the short time period. This process becomes more interesting when we convert the collected tweets into a text stream, and thus tweets contain information referring to numerous events spanning relatively diverse arguments. From the accuracy of the proposed clustering mechanism, we can observe that approach is performing well on very dedicated and focused events. The proposed approach shows better results on more complicated event text, including contextual information and heterogeneous events like the Enron mail stream. Therefore, we can say that proposed online fast clustering can extract more event-related topics than other methods. It also considers the event's contextual information, assigning a relevant number of clusters based on similarity. Figure 3 depicts the proposed approach's influence over accuracy and shows its variation concerning data size change in various text streams to detect events. Accuracy of the proposed clustering will be affected by increasing the text stream's size by which cluster size will also grow.

Furthermore, we explained the working of clusters. As we considered static and dynamic ways to understand the clustering on text streams, our approach shows the online clustering mechanism's benefits when it observes a new tweet in the text stream. Our research problem of detecting an event is eventually related to the classification of events. As we consider the online data stream clustering algorithm, the proposed approach sequentially extracts and processes the upcoming text stream. The mechanism will be in a continuous-time span. In this way,the whole stream of text

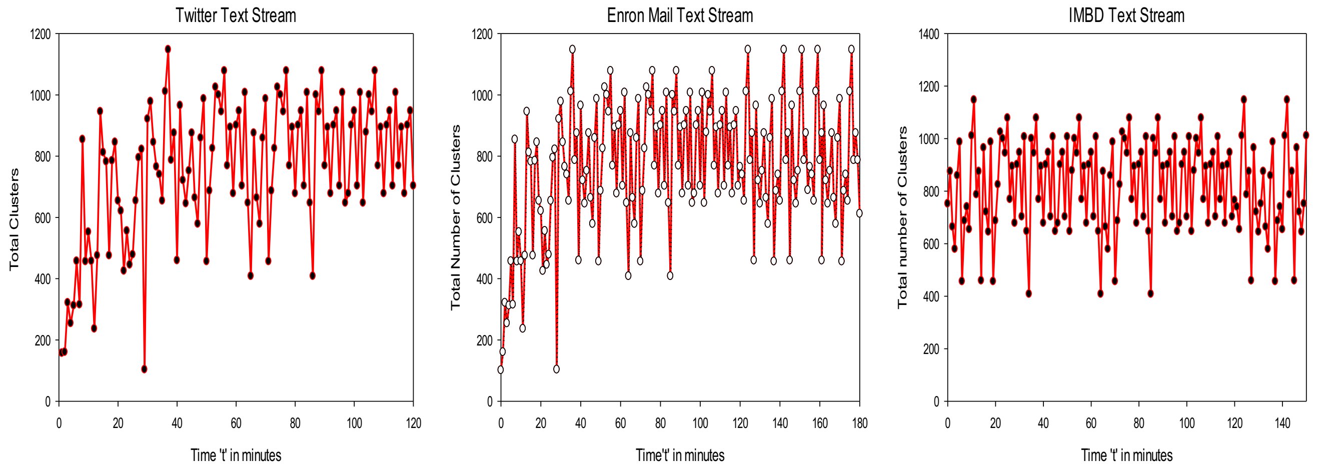

Fig. 4 Inspection of total number of clusters that turn into static/sedentary after a group of tweets on the basis of cluster capacity in various text streams

will be processed (one at a time) and incrementally updates clusters.

If any new tweet arrives,it will join the previously created cluster based on the tweet's similarity. The clusters' allocation process depends on the similarity to the newly arrived text; otherwise, if no similarity occurs, then a new cluster will be generated for the tweets. Thus, as time progresses, the arrival of a tweet reporting on a subject never appeared before it is deemed a new event. As in Fig. 4, clusters behavior is evaluated in different scenarios. It is observed that retrieving text from the stream a cluster plays a dual role,such as active and inactive, based on the upcoming text's nature. Therefore, Fig. 4 is elaborating the nature of clusters for different time periods. The cluster will accept the tweets if it's active; otherwise, it will never accept the event-related tweets or texts. It means that at different time span, the cluster becomes inactive and active. At the conclusion of this section, results show that the clusters which become static after a group of similar tweets of around 1000 are used for further processing. After becoming passive (static), it does not accept event keywords from the text stream. Even after becoming static for a longer duration, clusters' continuous functioning may only slow down the process of other event detection. Figure 5 shows the dynamic and static nature of clusters in the different time slots. For this purpose, we have divided the time into different time slots and compute the dynamic clusters in two ways. Firstly, we analyze the dynamic cluster in a particular time slot; secondly, we calculate the total time when the cluster is active.

6.5 Performance evaluation of the proposed approach with the increase of clusters and size of the text streams

The outcome of the various aggregations depends on the targeted algorithm, from the literature [33] huge the number of tweets aggregated, to have a fair estimate on the

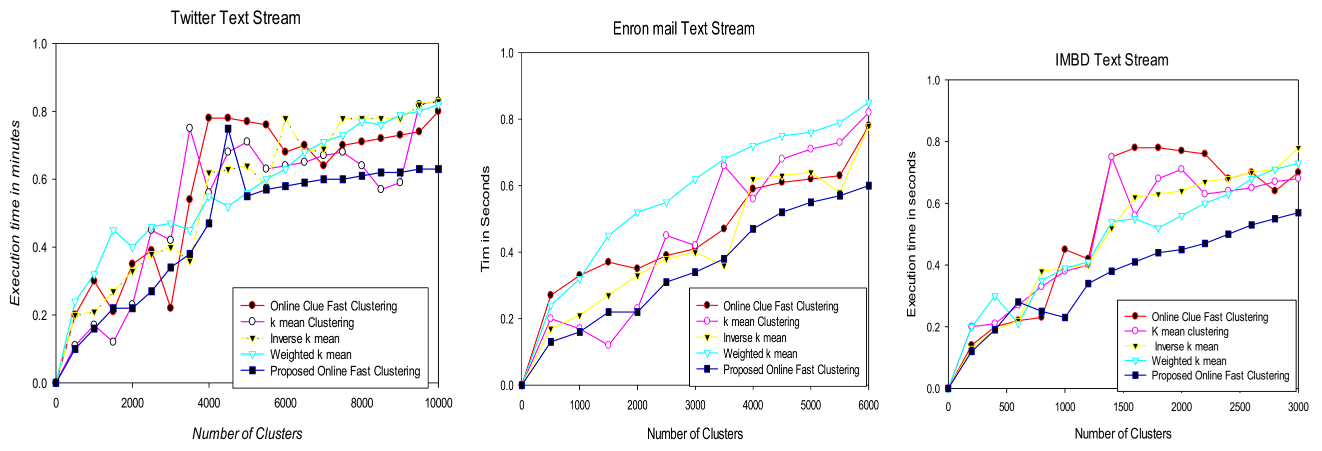

Fig. 5 Impact on execution time with increasing number of clusters, twitter text stream, enron mail stream and IMBD stream

recall value. The results show that the upcoming tweets' increase does not indicate that the number of clusters will also increase. Because a novel cluster will be generated only when a new event keyword is detected, it is observed that the increase of text stream affects overall performance. In the experiments, we first observed the impact of the increase in text size on the number of clusters using various standard and simulated datasets.This experiment found that the proposed clustering approach performs well and gives better results over the existing mechanisms. We tested our approach on all text streams used in this research work (Fig. 5), and the proposed approach is stable, reliable, accurate, and timekeeping than existing methodologies.

Furthermore, it is essential to check the performance of clustering efficiency with an increase in text stream size. The impact of size is directly associated with the performance of event detection and classification. During our experiment, we found that when the text stream's size increases,clustering performance degrades. The larger the data, the lower is the clustering performance,as the algorithm may miss a few keywords related to the events. Such type of situation can be critical when clusters ignore a large number of important information. Figure 6 depicts the results of various text streams. The results clearly indicate that the proposed approach is working well on an increased load of text.

6.6 Analyzing performance in fixed length and variable length of the text streams

Further, we extend our experimentation. We analyzed the performance of the proposed approach on the variation of clustering accuracy (event detection accuracy) concerning event size and response time with respect to the number of events.To train the Siamese Network, we followed the architecture and training method from [38]. For this purpose, we used a twin pair of LSTMs and implemented through Pytorch. In our research task, as we have converted text into embedding (compressed vector). Therefore, it would be impossible to compute cosine similarity or similarity measures to check the topical dissimilarity quotient between input and output. To capture this, we computed split error (inverse false positive indicator) and Jessen-Shannon(JS) Divergence on the given datasets. From Table 1/Fig. 7 (fixed-length text) and Table 2/Fig. 8 (variable length text), it is evident that the proposed scheme outperforms CNN and SVM-based methods towards the change in the input, i.e., sensitivity in event detection. From all the experiments, it can be discerned that our algorithms are robust towards the burstiness and heterogeneity in events present in text streams with sufficient sensitivity to detect early events.

7 The efficiency of the proposed event detection approach with the existing state of the art methodologies

We also investigated the proposed approach of event detection to analyze and compute the accuracy of events with varying time constraints. For this purpose,we initialized the probability of an event's occurrence to 8 minutes and resolute the epochs in which event equivalent or parallel to events feature keywords can be analyzed. In each case, clusters are assigned as explained above in Algorithm 2 to detect the events. The accuracy of the event detection algorithm is illustrated in terms of precision and recall, respectively. It is observed that in the detection of events,text features are not sufficient and accurate to gain a good estimate of efficiency. Therefore, other parameters, such as network structure and content of the text, also impact event detection efficiency. As we included different

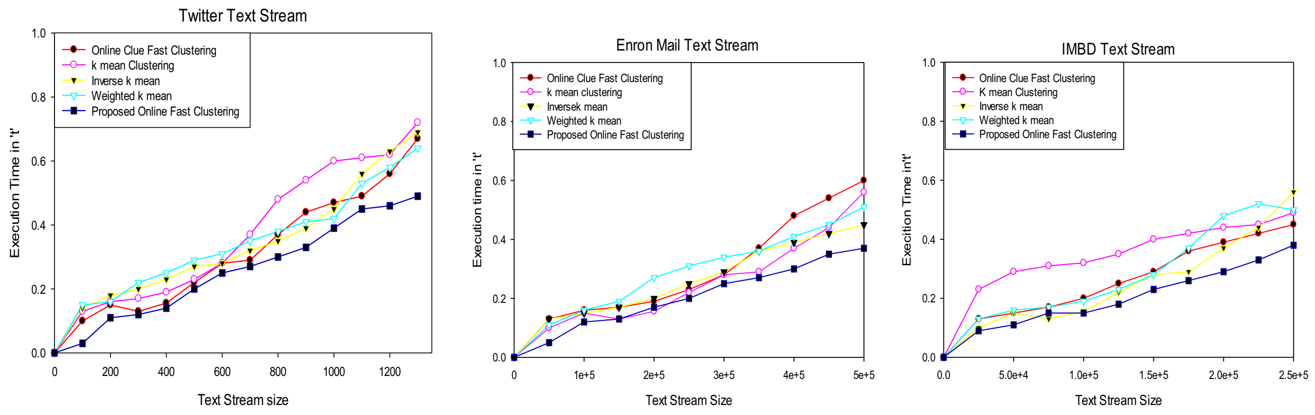

Fig. 6 Impact on execution time with increase the size of text stream, twitter text stream, enron mail stream and IMBD stream

Table 1 Results for various text stream using the fixed length of text

Approaches	Results for twitter stream		Results for enron mail stream		Results for IMBD text stream	
	Split error (%)	JS divergence	Split error (%)	JS divergence	Split error (%)	JS divergence
CNN	27.07	0.0765	30.12	0.0841	24.17	0.0912
VSM	34.71	0.0601	31.54	0.0512	25.84	0.0752
Proposed Scheme	34.81	0.0770	30.78	0.0823	24.23	0.0871

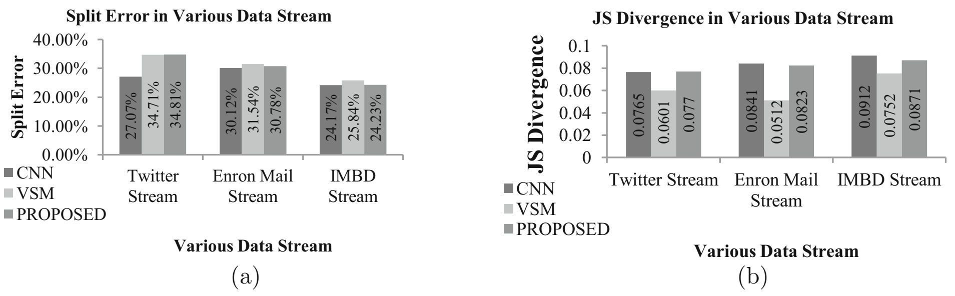

Fig. 7 **a** Depiction of split error and **b** Divergence and sensitivity depiction of various methods w.r.t fixed text streams

Table 2 Results for various text stream using embedding with variable length of text

Approaches	Results for twitter stream		Results for enron mail stream		Results for IMBD text stream	
	Split error (%)	JS divergence	Split error (%)	JS divergence	Split error (%)	JS divergence
CNN	32.29	0.3765	33.12	0.4841	31.57	0.3912
VSM	37.21	0.2601	35.54	0.1012	32.84	0.2752
Proposed Scheme	31.21	0.3670	33.78	0.4423	32.23	0.4171

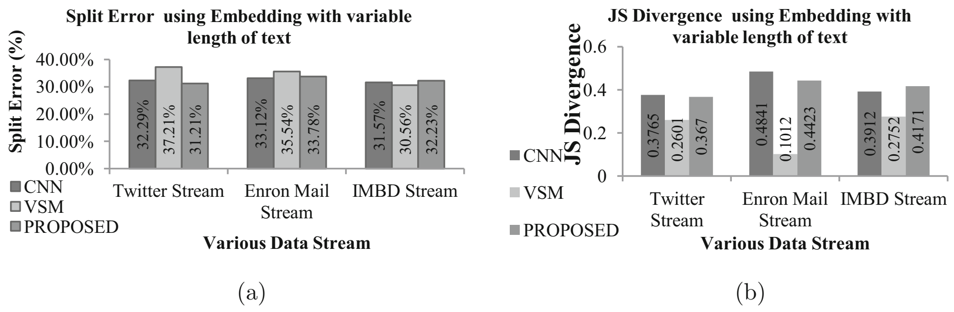

Fig. 8 **a** Depiction of split error and **b** Divergence and sensitivity depiction of various methods w.r.t variable text streams

social media text features in our experiment datasets, our proposed approach is stable to good efficiency in an accurate way, even in the noisy data situation (Table 3).

It is also observed that, the proposed model is giving promising results in terms of time complexity. We can say that the computation time of proosed approach is quite less than the standard baseline approaches as described in Table 4. Further, the accuracy of the proposed approach for event detection with its standard error is also calculated. Table 4 reveals the efficacy of the proposed approach. To compute the average accuracy of the proposed approach, we use five-fold cross-validation and a grid search procedure to identify and found that the proposed methodology behaves very well under various conditions Table 4.

7.1 Data sets and state of the art approaches

– *FA Cup final* FA cup or the Football Association challenge cup is a famous and mature society in the world. The first competition was held in 1871 and is the

Table 3 Efficiency of the proposed event detection algorithm

Data set	Baseline approaches for event detection	Precision	Recall	F1
FA CUP Final	C. Latent Dirichlet Allocation (LDA)	0.72	0.75	0.78
	Graph-based feature-pivot method topic discovery	0.78	0.77	0.80
	BN-grams	0.81	0.78	0.80
	Proposed Event Detection Approach	0.82	0.80	0.81
US elections	C. Latent Dirichlet Allocation (LDA)	0.77	0.73	0.75
	Graph-based feature-pivot method topic discovery	0.82	0.84	0.79
	BN-grams	0.79	0.83	0.85
	Proposed Event Detection Approach	0.81	0.84	0.87

chief competition in English football [13]. 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 the player's name. The ground reality included 13 topics such as start, middle, end of the match time, bookings and goals.

– *US elections* In this data set, information for the presidential election of 2012 is provided. Numerous featured keywords are used to extract and analyze text. Hashtags are used to describe the various features, and in the ground reality, 64 main topics are available. Tags are used to extract the event features related text from the dataset such as presidential race in a particular state, result, senate race result, and victory speech of Obama's [13]. Elections of the US magnetized enormously high levels of movement on twitter. Thus, numerous users tweet on the same topic and provide huge data to extract useful patterns. Furthermore, the proposed Methodology is evaluated, and the experiments are implemented to reveal effectiveness and efficiency. For this purpose, we compare our Methodology with three state of art approaches using three different data sets. This section discusses the existing approaches used in our experiment, which are explained in [13], such as LDA, graph-based, and bn-gram.

– *C. Latent dirichlet sllocation (LDA)* To address the topic based event detection probability based event detection methods are famous in addressing such problem domain in pure text-based datasets. Topic based models are the same as the Bayesian model where each and every document and feature keyword is linked with probability sharing over topics and turn allocation over topics. LDA is a very common and widely used model due to which reason we select it as a baseline to evaluate our approach beside. 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 which are hidden.

– *Graph-based feature-pivot topic discovery* This is another baseline approach that we used is to evaluate the efficiency of the event detection approach. To cluster the documents Structural Clustering Algorithm for Networks (SCAN) is used. This SCAN approach not

Table 4 Time complexity, average accuracy and standard errot of proposed approach w.r.t. existing approaches

Data set	Baseline approaches for event detection	Time complexity	Accuracy of the proposed approach (%)	Standard error
FA CUP Final	C. Latent Dirichlet Allocation (LDA)	0.24	86.54	0.07
	Graph-based feature-pivot method topic discovery	0.31	88.41	0.10
	BN-grams	0.27	89.49	0.09
	Proposed Event Detection Approach	0.17	93.32	0.04
US elections	C. Latent Dirichlet Allocation (LDA)	0.19	88.42	0.12
	Graph-based feature-pivot method topic discovery	0.22	89.65	0.11
	BN-grams	0.20	92.72	0.08
	Proposed Event Detection Approach	0.13	97.41	0.05

only perceives communities of nodes also provides 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 an unambiguous link between topics. Similar words can be cluster together on the basis of similarity, and the importance of the words can also be computed by considering the probability of words as $p(w|textstream)$ which includes event's feature selection, linking, clustering, and clustering enrichment [39].

– *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 together. In the third step, topic ranking is performed. Each tweet is to be normalized/pre-processed as for this purpose, we are using a similar approach again as described Algorithm 1. In this research work, we considered two aggregation typeson time and topics [39]. Furthermore, LSH is implemented on the collected tweets in a particular time slot based on tweet similarity.

8 Conclusion and future work

This research aims to present a methodology for event detection and classification in real-time scenarios at twitter and web-scale. We have explained the various steps to design the approach and evaluated its performance using different social text streams. As online clustering itself is a daunting task mainly when it concerns stream data, it is even more difficult in text streams emerging from social media, containing textual and symbols, images, abbreviations, and slangs, etc. Therefore, it is quintessential to employ some effective representation of input streamed data that converts heterogeneous input into synchronous and unified form and handle the difficulty of expressing semantics in numeric values. Our approach presents an embedding based (Word2vec based) input representation obtained after normalization to cater to this challenge. The next crucial issue is to detect and classify unique/ novel events in the input stream; for this, we proposed one-shot learning methods using Siamese Neural Networks to differentiate between evolutionary vs. novel events with early detection. One of our work's significant achievements is its sensitivity towards false positive events, which not only filters out the false event but also ranks events based on their influences. In the future, the proposed scheme can experiment on pure images and video data streams.

Funding The authors have not disclosed any funding.

Data availability Enquiries about data availability should be directed to the authors.

Declarations

Conflict of interest The authors have not disclosed any competing interests.

References

1. Von Nordheim, G., Boczek, K., Koppers, L.: Sourcing the sources: an analysis of the use of twitter and facebook as a journalistic source over 10 years in the new york times, the guardian, and süddeutsche zeitung. Digit. J. **6**(7), 807–828 (2018)
2. Imran, M., Castillo, C., Diaz, F., Vieweg, S.: Processing social media messages in mass emergency: a survey. ACM Computi. Surv. (CSUR) **47**(4), 1–38 (2015)
3. Xie, W., Zhu, F., Jiang, J., Lim, E.P., Wang, K.: Topicsketch: real-time bursty topic detection from twitter. IEEE Trans. Knowl. Data Eng. **28**(8), 2216–2229 (2016)
4. 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. In: Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, pp. 513–522 (2016)
5. Zhang, C., Liu, L., Lei, D., Yuan, Q., Zhuang, H., Hanratty, T., Han, J.: Triovecevent: Embedding-based online local event detection in geo-tagged tweet streams. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 595–604 (2017)
6. Alsaedi, N., Burnap, P., Rana, O.: Can we predict a riot? disruptive event detection using twitter. ACM Trans. Internet Technol. (TOIT) **17**(2), 1–26 (2017)
7. Kumar, J.P., Govindarajulu, P.: Near-duplicate web page detection: an efficient approach using clustering, sentence feature and fingerprinting. Int. J. Comput. Intell. Syst. **6**(1), 1–13 (2013)
8. Barbakh, W., Fyfe, C.: Online clustering algorithms. Int. J. Neural Syst. **18**(03), 185–194 (2008)
9. Sumalatha, M., Ananthi, M.: Efficient data retrieval using adaptive clustered indexing for continuous queries over streaming data. Cluster Comput. **22**(5), 10503–10517 (2019)
10. Wei, C.P., Lee, Y.H., Hsu, C.M.: Empirical comparison of fast partitioning-based clustering algorithms for large data sets. Expert Syst. Appl. **24**(4), 351–363 (2003)
11. Jiang, X., Zhang, N., Huang, J., Zhang, P., Liu, H.: Analysis of prediction algorithm for forest land spatial evolution trend in rural planning. Cluster Comput. 1–9 (2021)
12. Vavliakis, K.N., Symeonidis, A.L., Mitkas, P.A.: Event identification in web social media through named entity recognition and topic modeling. Data Knowledge Eng. **88**, 1–24 (2013)
13. 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 Trans. Multimedia **15**(6), 1268–1282 (2013)
14. Banda, L., Bharadwaj, K.K.: An approach to enhance the quality of recommendation using collaborative tagging. Int. J. Comput. Intell. Syst. **7**(4), 650–659 (2014)
15. Cadenas, J.M., Garrido, M.C., Martínez, R.: Nip-an imperfection processor to data mining datasets. Int. J. Comput. Intell. Syst. **6**(sup1), 3–17 (2013)

16. Hasan, M., Orgun, M.A., Schwitter, R.: A survey on real-time event detection from the twitter data stream. J. Inf. Sci. **44**(4), 443–463 (2018)
17. Weiler, A., Grossniklaus, M., Scholl, M.H.: Survey and experimental analysis of event detection techniques for twitter. Comput. J. **60**(3), 329–346 (2017)
18. Yao, J., Cui, B., Xue, Z., Liu, Q.: Provenance-based indexing support in micro-blog platforms. In: 2012 IEEE 28th International Conference on Data Engineering, pp. 558–569. IEEE (2012)
19. Singh, T., Kumari, M.: Role of text pre-processing in twitter sentiment analysis. Procedia Comput. Sci. **89**, 549–554 (2016)
20. Singh, T., Kumari, M., Pal, T.L., Chauhan, A.: Current trends in text mining for social media. Int. J. Grid Distrib. Comput. **10**(6), 11–28 (2017)
21. Aggarwal, CC., Subbian, K.: (2012) Event detection in social streams. In: Proceedings of the 2012 SIAM international conference on data mining, SIAM, pp. 624–635
22. Xu, Q., Li, M.: A new cluster computing technique for social media data analysis. Clust. Comput. **22**(2), 2731–2738 (2019)
23. Dong, X., Mavroeidis, D., Calabrese, F., Frossard, P.: Multiscale event detection in social media. Data Min. Knowl. Disc. **29**(5), 1374–1405 (2015)
24. Wang, Z., Shou, L., Chen, K., Chen, G., Mehrotra, S.: On summarization and timeline generation for evolutionary tweet streams. IEEE Trans. Knowl. Data Eng. **27**(5), 1301–1315 (2014)
25. Camacho-Collados, J., Pilehvar, M.T.: From word to sense embeddings: A survey on vector representations of meaning. J. Artif. Intell. Res. **63**, 743–788 (2018)
26. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. (2013) arXiv preprint arXiv:1301.3781
27. Becker, H., Naaman, M., Gravano, L.: Beyond trending topics: Real-world event identification on twitter. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 5, (2011)
28. Fedoryszak, M., Frederick, B., Rajaram, V., Zhong, C.: (2019) Real-time event detection on social data streams. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pp. 2774–2782
29. Hasan, M., Orgun, M.A., Schwitter, R.: Real-time event detection from the twitter data stream using the twitternews+ framework. Inform. Process. Manag. **56**(3), 1146–1165 (2019)
30. Jiang, Z., Gao, S.: An intelligent recommendation approach for online advertising based on hybrid deep neural network and parallel computing. Clust. Comput. **23**(3), 1987–2000 (2020)
31. Yadav, A., Vishwakarma, D.K.: A comparative study on bio-inspired algorithms for sentiment analysis. Clust. Comput. **23**(4), 2969–2989 (2020)
32. De Boom, C., Van Canneyt, S., Demeester, T., Dhoedt, B.: Representation learning for very short texts using weighted word embedding aggregation. Pattern Recogn. Lett. **80**, 150–156 (2016)
33. Tang, J., Qu, M., Mei, Q.: Pte: Predictive text embedding through large-scale heterogeneous text networks. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1165–1174 (2015)
34. Jiang, Y.: Semantically-enhanced information retrieval using multiple knowledge sources. Cluster Comput. 1–20 (2020)
35. Pimentel, M.A., Clifton, D.A., Clifton, L., Tarassenko, L.: A review of novelty detection. Signal Process. **99**, 215–249 (2014)
36. Karkali, M., Rousseau, F., Ntoulas, A., Vazirgiannis, M.: Efficient online novelty detection in news streams. In: International conference on web information systems engineering, pp. 57–71. Springer(2013)
37. Zhao, H., Gallo, O., Frosio, I., Kautz, J.: Loss functions for image restoration with neural networks. IEEE Trans. Comput. Imaging **3**(1), 47–57 (2016)
38. Mueller, J., Thyagarajan, A.: Siamese recurrent architectures for learning sentence similarity. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30 (2016)
39. Winarko, E., Pulungan, R., et al.: Trending topics detection of indonesian tweets using bn-grams and doc-p. J. King Saud Univ.-Comput. Inform. Sci. **31**(2), 266–274 (2019)

Tajinder Singh is working as a Assistant Professor at Sant Logowal Institute of Engineering and Technology, Longowal, Punjab India. He completed his Ph.D entitled "Machine Learning Based Text Mining in Social Media" from National Institute of Technology, Hamirpur. His research interests are in the broad area of Text Mining and Machine Learning including their semantics, concurrency theory, event detection, classification, burst event identification of social media and social networks including formal foundations of networks especially in text analytics including formal stream flow models for online social media real time analysis. Dr. Singh also has some work in data science and digital image processing. In the broad approach to text analytics issues, some of the areas have explored in which Dr. Singh have guided master and graduated students include security and integrity of information, sentiment analysis, and Semantic representation of social media text and encoding of social media text. Before joining SLIET Longowal, Sangrur, Dr. Singh was working as an Assistant Professor at Indian Institute of Management Kashipur, Uttrakhand. Dr. Singh has also international experience of teaching as he has served as an Assistant Professor in the University of Information Sciences and Technology, Ohrid, Macedonia. His publications have appeared in peer-reviewed journals including, International Journal of Grid and Distributed Computing, Computer Communication (Elsevier, Journal of Super Computing (Springer), Expert Systems (Elsevier)), and has presented his research at several national and international conferences.

Madhu Kumari earned her M.Tech. and Ph.D. degree from Jawaharlal Nehru University, New Delhi INDIA, during which she researched on various aspects of Machine Learning algorithms in Computational Advertising, Social Network Analysis and Data Science. She is an active reviewer in peer reviewed journals of publishers like Elsevier, Taylor and Francis, Springer and Inderscience. Currently she is working at University Center of Research and Development in Chandigarh University, Gharuan, Punjab, India.

Daya Sagar Gupta received the B. Tech. degree in Computer Science and Engineering from UPTU Lucknow (UP), India in 2011, and M. Tech. degree in Computer Science and Engineering with distinction from Indian Institute of Technology (Indian School of Mines) [IIT (ISM)] Dhanbad, Jharkhand, India in 2014. He received the Ph.D. degree from IIT (ISM) Dhanbad, India in 2018. Presently, he is working as an Assistant Professor in the Department of Computer Science and Engineering, Rajiv Gandhi Institute of Petroleum Technology Amethi, UP, India. He published a number of research papers in reputed journals and conferences. He is also serving as reviewer in many reputed journals and served as a PC member in many international conferences. His research interest includes Information Security, Lattice-based Cryptography, Blockchain, Machine Learning and IoT Security.