Expert Systems With Applications 189 (2022) 116086

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

An evolutionary event detection model using the Matrix Decomposition Oriented Dirichlet Process

P.M.A. Yashar Erfanian [a], Bagher Rahimpour Cami [a], Hamid Hassanpour [b,*]

[a] Faculty of Computer Engineering and Information Technology, Mazandaran University of Science and Technology, P.O. Box 734, Babol, Iran
[b] Faculty of Computer Engineering and Information Technology, Shahrood University of Technology, P.O. Box 316, Shahrood, Iran

ARTICLE INFO

Keywords:
Event detection
Event evolution
Topic modeling
Social network analysis
Incremental clustering

ABSTRACT

With the huge expansion of user generated content on social networks, event detection has emerged as a major challenge and source of knowledge discovery. This knowledge is employed in different applications such as recommender systems, crisis management systems, and decision support systems. Dynamicity, overlapping, and evolutionary behavior are the most important issues in event detection. This paper proposes a novel evolutionary model for event detection to capture the dynamism and evolving behavior of events. The proposed method uses a matrix decomposition technique and a Dirichlet Process to detect events and handle their dynamicity. This model consists of two components, namely preliminary event detection and event evolvement tracking. The former component extracts preliminary events from the available data using the matrix decomposition method. Then, subsequent data is employed into a Non-Parametric Bayesian Network, namely Dirichlet Process Mixture Model to evolve the preliminary events. During the evolvement process, data may migrate between extracted events or new events may be discovered. The experimental results and comparisons with several recently developed approaches show the superiority of the proposed approach, and its ability to capture the evolutionary behavior of events over time.

1. Introduction

In recent years, social media networks, such as Twitter and Facebook, have provided a framework for information dissemination and social interactions in real time among their users (Shi et al., 2017). These frameworks provide interactive technologies that allow users to share their activities, opinions and interests with other people in real time. In other words, they operate as sensors in a large geographic area to inform, discuss and exchange information about product announcements, natural disasters, and any real-life events (Cordeiro & Gama, 2016). Thus, social media have provided a rich resource of information. Meanwhile, event detection has emerged as a major issue in social network analysis to discover useful knowledge (Lu et al., 2019).

An event conventionally represents a collection of posts (documents) with a common content showing a collective interest in the social network. Each event is usually specified by a topic, time, and its associated entities (Dou et al., 2012). For example, the *"Coronavirus disease 2019 (COVID-19) spread"* (as event topic) is extensively reported on Twitter where users are discussing economic issues (as associated entity) in 2020 (as time). Event detection systems discover events from huge data extracted from social media. The knowledge discovered from event detection systems can be used in various areas such as crisis management systems (Alsaedi & Burnap, 2015; Pohl et al., 2015; Sakaki et al., 2010), decision support systems (Zhou & Chen, 2014), recommender systems (Cui et al., 2017), news events detection (Hasan et al., 2019; Hu et al., 2017), and disease spreading (Cui et al., 2017).

In literature, there are three approaches for event detection, namely feature-based (Guille & Favre, 2015), topic-modeling-based (Chen et al., 2018; Xing et al., 2016), and incremental (Hasan et al., 2019) approaches. Feature-based approaches use document features such as text, time, and hashtags. Topic-modeling-based approaches use probabilistic methods to find latent factors as events. These approaches assume that the number of events stays fixed over time (Capdevila et al., 2018; Xing et al., 2016; Zhao et al., 2011). Thus, they cannot capture the growth of events over time and discover new events. To overcome this problem, incremental approaches have been developed which use an incremental clustering such as Dirichlet process to detect newly emerged events (Bandaragoda et al., 2017; Hasan et al., 2019).

The main issue in event detection is to discover actual events in real world by processing streaming data on social media and tracking them along time. An event detection system is expected to extract new events and track them over time. Although, existing methods do focus on these two subjects, but there are two main challenges which include

* Corresponding author.
E-mail addresses: yapmayashar@gmail.com (P.M.A.Y. Erfanian), rc_bagher@yahoo.com (B.R. Cami), h.hassanpour@shahroodut.ac.ir (H. Hassanpour).

https://doi.org/10.1016/j.eswa.2021.116086
Received 20 June 2020; Received in revised form 23 June 2021; Accepted 12 October 2021
Available online 21 October 2021

discovering events and tracking their subsequences. First, it is necessary to predefine the expected number of events in feature-based and topic modeling-based approaches, in advance. However, it is difficult to accurately estimate the number of events for a given streaming social data. For example, the online Latent Dirichlet Allocation (OLDA) (Hoffman et al., 2010) method requires to fix the number of events in advance. Also, Sampled Min-Hashing (SMH) (Fuentes-Pineda & Meza-Ruiz, 2019) uses an algorithm to determine the number of events, but its limitation is the fixing of events over time. Second, methods which have an incremental approach focus on emerging events and tracking them but the content (documents) of events are fixed over time. For example, authors in Chen et al. (2018) introduced the RL-LDA model which used hashtag, location, text, and retweeting behavior to capture events. The RL-LDA model incorporates a dynamic update algorithm, which incrementally updates events over streaming data. Also, Multimodal Event Topic Model (mmETM) (Qian et al., 2015) incorporated an incremental learning strategy to event tracking and was adopted for tracking social events. These methods focused on embedding events over time but do not capture the dynamicity of event content.

Overall, the existing methods focus on event extraction and handling emerging events. These methods do not capture event content evolution. Due to evolutionary behavior, events not only appear over time but also their content can be changed and enriched over time. For example, several months before the Olympics, its related news are shared by social media users. Thus, the *Olympic event* is detected by event detection systems. Once the Olympic starts, the Olympic event are decomposed into new events such as *the champions* and *the participating countries*.

In this paper, we propose an evolutionary model for event detection. The proposed method employs the Matrix Decomposition Oriented Dirichlet Process for developing an Evolutionary Event Detection Model (EEDM) which includes two components, namely preliminary event detection and event evolvement tracking. The preliminary event detection component incorporates the Non-negative Matrix Factorization (NMF) algorithm (Lee & Seung, 1999) and a probabilistic function for discovering events from the available documents (data). The evolvement tracking component employs timely progressing documents (data) and applies them to the preliminary events. Also, it is able to identify an infinite number of events over time using the distance dependent Chinese Restaurant Processes (dd-CRP) algorithm (Blei & Frazier, 2011). In addition, in the evolvement tracking component, documents may be migrated (reassigned) from an event to the most relevant event over time, hence the purity of each event is enhanced.

We use TwitterBreakingNews (Kalyanam et al., 2016), 20NewsGroup, and Grolier datasets to evaluate the proposed method. Also, it was compared with recently developed methods which were selected from the aforementioned approaches. The empirical results confirm the performance of the proposed method at detecting and tracking events on social media.

The main contribution of this paper is to introduce an evolutionary event detection model which is able to extract events and discover emerging events on social streaming data. The proposed method captures the dynamic content of events along with emerging new data over time.

The remainder of this paper is organized as follows. Section 2 provides a brief review on the existing event detection methods. The design details of the proposed method components are presented in Section 3. Section 4 provides the experimental results. In Section 5, we present our conclusions and future work.

2. Related works

There are diverse methods for detecting events from social media. The existing methods have been used to discover significant information on various types of events such as earthquakes, epidemics, traffic monitoring, and elections (Cui et al., 2017; Goswami & Kumar, 2016; Sakaki et al., 2010; Unankard et al., 2014). As mentioned earlier, current methods for event detection can be divided into three categories, namely feature-based, topic-modeling-based, and incremental approaches. These approaches are discussed below.

Feature-based approaches. In this approach, event detection algorithms employ the features of documents such as textual keywords, timestamp, and hashtags to discover events (Chen et al., 2018). Also, some methods (Capdevila et al., 2017, 2018) have focused on geographical location-based event identification.

The authors in Guille and Favre (2015) presented a Mention-Anomaly-Based Event Detection (MABED) which employs keywords of tweets into a statistical technique to discover events. Also, in Zuo et al. (2016) keywords were employed in a weighted graph based on word co-occurrence to discover events. The authors in Zhang et al. (2016), in addition to the keywords, used the location of tweets to find local events, and rated the identified events to monitor and capture the events. The authors in Gaglio et al. (2016) and Stilo and Velardi (2016) employed timestamp data in their event detection algorithms. in Stilo and Velardi (2016), temporal co-occurrence was used to solve the problem of tweet shortness. To this end, the Symbolic Aggregate approXimation (SAX) (Lin et al., 2007) technique was used to transform word temporal series into a string of symbols. In the Neural Variational Document Model (NVDM) paper (Miao et al., 2016) the authors provide a form of multinomial matrix factorization framework for generative models of texts using variational autoencoder. The Gaussian Softmax distribution (GSM) (Miao et al., 2017) is a neural modeling algorithm which constructs a finite topic distribution followed by NVDM. Also, in Gaglio et al. (2016) a dynamic time window was employed in an event detection algorithm, named Twitter Live Detection Framework (TLDF), to track events over time. The variable length of the time window made it possible to match the event length based on the actual volume of tweets at the time of occurrence.

This approach assumes that the number of events is constant. However, due to the dynamic nature of social network environments, it is not applicable to choose a constant number of events in consecutive constant time intervals.

Topic-modeling-based approaches. They utilize probabilistic models such as topic modeling (Blei et al., 2003; Hofmann, 1999) for event detection, and discover the distribution of latent topics over documents. Then, each topic distribution is considered as an event. In other words, events are considered as latent variables which are learned from documents corpus. Latent Dirichlet Allocation (LDA) (Blei et al., 2003) is the most known probabilistic topic modeling technique. There are various methods that utilize LDA as a basis for event detection (Cai et al., 2015; Chen et al., 2018; Diao et al., 2012; Hoffman et al., 2010; Xing et al., 2016; Zhao et al., 2011). In the following, we present a number of well-known methods which utilize different extensions of LDA for event detection.

The authors in Xing et al. (2016) extracted hashtags from tweet documents and applied LDA on them. They provided a topic distribution to represent events. In another LDA-based model called RL-LDA (Chen et al., 2018), the model employs some useful features such as textual content, time, location, hashtag, and retweets of users for event detection. The real-time RL-LDA model used the retweet feature to handle the generation of new events over time. TimeUserLDA (Diao et al., 2012) is an offline probabilistic event detection model which has two main components. The first component presents a topic model which separates personal topics from general topics according to temporal information. The second component uses the Poisson state machine to detect bursty topics over a period of time. In Twitter-LDA (Zhao et al., 2011), a probabilistic model was introduced to overcome the problem of discovering short text topics (Unankard et al., 2015). The Twitter-LDA algorithm models each tweet as a combination of unigram words. In this probabilistic model, only one topic is assigned per tweet. The ProdLDA (Srivastava & Sutton, 2017) algorithm is an advanced version

of the traditional LDA algorithm that uses expert generated vocabulary instead of a mixture model.

Further, numerical, textual, and multimedia contents can also be used in event detection methods. The authors in Cai et al. (2015) considered five features, including text, location, image, hashtag, and history to construct a probabilistic model for event detection. in Cai et al. (2015), each topic was a combination of distributions over hashtag, words, time, and images. The Spatio-temporal Multimodal TwitterLDA (STM-TwitterLDA) method (Cai et al., 2015) uses a Convolutional Neural Network (CNN) rather than the traditional bag-of-visual-words method (Yang et al., 2007) to employ images. It used a graph based technique called Maximum-Weighted Bipartite Graph Matching (MWBGM) (Long et al., 2011) to capture event tracking by linking events between consecutive days. The Multi-modal Event Topic Model (mmETM) (Qian et al., 2015) is able to identify events via relating visual and non-visual topics. This model is able to track events by calculating the similarity between events over time.

The authors in Lu et al. (2019) used the non-parametric Recurrent Chinese Restaurant Process (RCRP) model (Ahmed & Xing, 2008) for detecting and tracking events over time. Also, it utilized the Long Short-Term Memory (LSTM) model (Hochreiter & Schmidhuber, 1997) to learn short-term and long-term semantic dependencies of the evolving trends. The authors in Huang et al. (2015) presented an evolutionary method called Event Evolution Mining (EEM) to identify short text events on the web (the evolution of an event signifies how its popularity changes over time). The EEM used the Finding Topic Clusters using Co-occurring Terms (FTCCT) technique and performed the Co-occurring Terms clustering for Finding Topic Clusters. The Sampled Min-Hashing (SMH) (Fuentes-Pineda & Meza-Ruiz, 2019) algorithm is able to detect latent topics from massive texts. Although SMH determines the number of events, but it is unable to detect new clusters over time. in Wang et al. (2019) the authors proposed a neural topic model based on Generative Adversarial Nets (GANs) named Adversarial Topic Model (ATM), which uses generator network to capture semantic patterns between latent topics. The Warble algorithm (Capdevila et al., 2018) detected events according to their geographical locations. It utilized a spatio-temporal probabilistic method to reduce the gap between virtual and real-world social networks.

Generally, in the above-mentioned methods, the number of events must be predetermined. Due to the dynamic behavior of events in social media, a major issue in event detection is to capture emerging and evolving events. Thus, incremental approaches have been developed in event detection to overcome the problem of assuming a constant value for the number of events. In the following, we further discuss the incremental algorithms.

Incremental approaches. Due to the real-time nature of social media and continuous interaction between online social network participants, events continuously occur over time. Thus, the incremental approach not only performs event detection but also discovers upcoming events from stream data. In other words, this approach utilizes methods to handle event tracking and capture emerging events. In the following, we present some methods which take an incremental approach towards event detection.

The authors in Petrović et al. (2010) used the nearest neighbor search to construct an event detection system called First Story Detection (FSD). The nearest neighbor algorithm gathers similar documents into groups, and then each group is considered as an event. They utilized the Locality Sensitive Hashing (LSH) method (Indyk & Motwani, 1998) to speed up the search in nearest neighbors. Detected events are sorted according to a novelty measure (Ibrahim et al., 2018) and the high score events are considered as hot events. The authors in Hasan et al. (2019) introduced a real-time and incremental event detection framework called TwitterNews+ for detecting news events from Twitter. They used two modules, namely search and event detection where the search module removes duplicate tweets, while the second module employs a vector space model of new tweets into an incremental clustering to discover events. Commonly, event detection methods utilize the Term Frequency–Inverse Document Frequency (TF–IDF) (Chowdhury, 2010) statistical technique for employing document corpus. The authors in Hu et al. (2017) believed that TF–IDF does not take into account the semantic relationships between words. To reduce the problems of large data dimensions and sparse semantics of the TF–IDF method, they introduced an event detection method based on word embeddings. Also, they introduced an online clustering method to identify news events.

The authors in Wang et al. (2013) assumed that topics can be divided into groups of identical tweets with specific time tags. They applied a Gaussian frequency distribution on a vector space model of the document corpus. They introduced Time-Dependent Hierarchical Dirichlet Process (TD-HDP) model which integrates the Hierarchical Dirichlet Process (HDP) and timestamp of documents. In other words, the Dirichlet Process (DP) is applied to the documents of each interval. They utilized the output of each interval and performed event tracking. The authors in Srijith et al. (2017) applied the HDP technique and discussed sub-events detection and event overlapping. The authors in Xie et al. (2016) utilized sketch-based topic modeling and introduced a real-time event detection method called TopicSketch. They defined two features named velocity and acceleration for topics. The velocity and acceleration of the topics were calculated and used to identify bursty events in the shortest possible time. Some researchers have used signal processing techniques for event detection (Nguyen & Jung, 2015, 2017). For example, the authors in Nguyen and Jung (2015) used vector space model of corpus of documents to construct the corresponding signal via keywords frequency. They employed user behavior factors such as retweeting and propagating in signal modeling. Signal processing techniques are used to detect spikes and anomalies, and consider them as events. The authors in Bandaragoda et al. (2017) separated the topic modeling process from event detection. They divided documents into separate time intervals with the interval value being determined apriori. Then, an incremental algorithm called Incremental Knowledge Acquisition and Self Learning (IKASL) (De Silva & Alahakoon, 2010) is applied to these pieces of data. The related topics in different time intervals are then linked together like a chain. This chain is referred to as the topic route, which is calculated for each of the topic clusters in the scoring route based on the sensitivity of the topic cluster. If the topic cluster score exceeds the threshold, it will be a candidate event.

In addition to spatio-temporal event detection, the authors in Dong et al. (2015) believe that events do not always occur in fixed time intervals and geographical distances. They used graph-based clustering such that tweets are the vertices and the similarities between tweets are the edges of the graph. For similarity measure calculation, a wavelet technique was applied to the temporal, spatial, and textual features of the tweets. The time resolution and location resolution are the main factors of the similarity measure. Thus, if two tweets are similar in content and temporally close to each other, their spatial distance will be ignored. Also, if they are similar in content and spatially close to each other, their temporal distance will be ignored.

Although existing methods are able to recognize new events which appear over time, they cannot capture the adaptive content of events. Adaptive content results in dynamic behavior of events in order to increase the purity of the event. In this paper we introduce an evolutionary event detection model to handle the dynamic behavior of events and their evolvement over time.

3. The proposed method

In this paper, we propose a method to detect events from streaming textual documents from social media. We consider the event evolution phenomenon in our event detection. Hence, not only new events are detected but also already detected events are purified via migrating the documents between the events over time. The migrations reduce

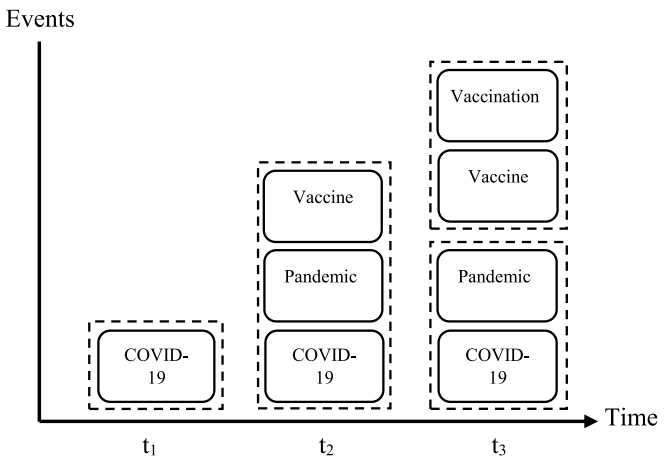


Fig. 1. Event evolution over time.

Table 1
Notations used to represent proposed method.

Notation	Description
D	Document corpus
d_i	ith document of corpus R
N	Number of documents (the size of D)
E	Events set
E_k	kth event of events set
$d_i^{E_k}$	ith document that located at the kth event
E_{d_i}	ith document event
$\theta(E_k)$	Priority of kth event
α	Dirichlet prior
γ	The threshold parameter for selecting document neighbors

the noisy content of events. For example, if a message with little topic relevance is assigned to an event, it can be migrated to a newer and more-relevant event over time. As depicted in Fig. 1, at time t_1 the most considerable event was *COVID-19*. At t_2 *Vaccine* was appeared along with COVID-19. During the time, by emerging new data, events COVID-19 and Vaccination are constructed as two disjoint events at time t_3. Accordingly, we propose an event detection model to capture the evolutionary behavior of events over time. Fig. 2 illustrates the overall architecture of the proposed method. Each component is described in this section.

For the proposed method, we consider a social media platform such as Twitter in which data items are created from streams of textual documents. Since social data are constantly streaming, the proposed method consists of two components, namely preliminary event detection and event evolvement tracking. Also, we define an event as a collection of documents from social media which have a common content. Commonly, an event may have a timestamp or may occur periodically.

We consider a vector space model of text documents as a data item of social media. A text document d_i, consists of a vector of words $W = \{w_1, w_2, \ldots, w_n\}$. Notation D represents a corpus which contains documents. The size of corpus is denoted with N. An event E_k is defined as a group of similar documents from corpus. Thus, $d_i^{E_k}$ represents the ith document of kth event. We measure the significance of E_k using function $\theta(E_k)$. Notation γ shows a threshold parameter for document neighborhood which is used in evolution process. A short description of notations is provided in Table 1.

The overall view of the proposed method is represented in Algorithm 1. In this algorithm, the *PreliminaryEventDetection*() function employs NMF and a probabilistic function to discover the preliminary events from initial data as *events*. Then we apply new data to these preliminary *events* and update *events* via discovering new events or keep the content adaptivity of *events*. Therefore if a new data d^* arrived, Event-Detection-Step performs event detection incrementally. To this end, Event-Detection-Step draws the appropriate destination event e from *events* for d^* or creates a new event and updates *events*. Finally, Event-Evolution-Step performs event evolution on *events* to enrich content adaptivity of *events*. In other words, Event-Evolution-Step performs content migration of events to handle the dynamic behavior of events. This migration is performed on neighbors of new data (d^*).

3.1. The preliminary event detection component

The preliminary event detection component is used to extract events from initial data. To this end, pre-processing operations such as tokenizing, normalizing, stop words removing, and lemmatizing are applied to the data (Hasan et al., 2019). We defined an event as a collection of similar documents considering their containing words. Therefore, we tokenized each document by using a uni-gram approach and removed any stop words and URLs. Because of our datasets (TwitterBreakingNews Kalyanam et al., 2016, 20NewsGroup,[1] and Grolier[2]) which are news-wire texts, we used standard tokenizer and morphological analysis (lemmetizing and stemming) tools to perform pre-processing. In this paper we focused on event evolution, not the content of event. Therefore content-wise operations such as NER[3] and POS[4] are not conducted. After identifying lemma of words in pre-processing phase, the cleaned data (documents with their all accepted words) are converted to the appropriate format (weighted matrix) which is required for the event extraction process using TF–IDF (Chowdhury, 2010) algorithm. Then, the NMF (Lee & Seung, 1999) matrix decomposition method is applied to the weighted matrix. NMF decomposes the weighted matrix into *document* × *topic (H)* and *topic* × *word (Z)* matrices. The former consists of topic weighted vector per each document, while the latter is composed of words weighted vector per each topic. Most of existing methods, such as Chen et al. (2019) used an argmax(H) function for documents clustering, but we utilize a probability function. The probability function uses each row of H as a weighted vector M_{d_i}. The weighted vector M_{d_i} represents the similarity between document d_i and the corresponding topics. This function which is brought in Eq. (1), find the destination event of d_i based on similarity vector values. This process is depicted in Fig. 3.

$$p(Event_{d_i} = E_k | M_{d_i}) = \underset{E_k}{\operatorname{argmax}}(M_{d_i} \times AvgSim(d_i, d_{-i})_{d_{-i} \in E_k}) \quad (1)$$

Finally, documents with similar topics are clustered into a specific group. We consider each group as an event. Algorithm 2 shows the preliminary event detection process. After extracting events from the initial data, we connect documents of an event to construct a graph (Blei & Frazier, 2011). In other words, a graph $G(V, E)$ is created from documents of an event where V and E denote the documents set and pairwise similarity, respectively. This graph is used in the event evolvement tracking component.

3.2. The event evolvement tracking component

The event evolvement tracking component handles the dynamic behavior of events and constructs new events if any. Thus, once a new document arrives, it goes through two phases. At first, the document is assigned to an existing event, or a new event is generated. Then, to capture the dynamic behavior of the events, the document

[1] http://qwone.com/~jason/20Newsgroups.
[2] https://cs.nyu.edu/~roweis/data.
[3] Named-Entity-Recognition. https://medium.com/mysuperai/what-is-named-entity-recognition-ner-and-how-can-i-use-it-2b68cf6f545d.
[4] Part-Of-Speech tagger. https://nlp.stanford.edu/software/tagger.shtml.

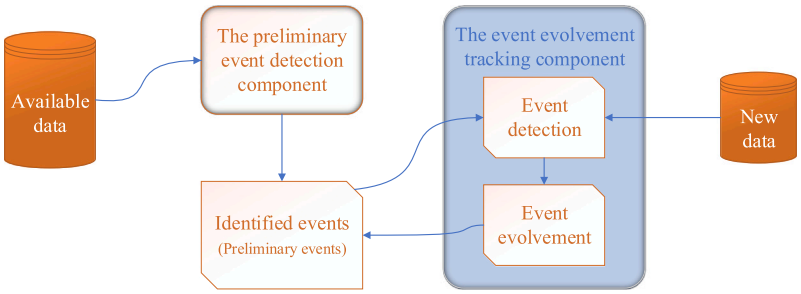


Fig. 2. Architecture of the proposed method.

Algorithm 1 The overall view of the proposed method

```
procedure EEDM
    Inputs: D ← Documents corpus of the microblog (Twitter)
    events = { }
    documents = PreProcess(D)
    events = PreliminaryEventDetection(documents)
    while true do
        d* = GetNewDocument()
        Event-Detection-Step:
                assign d* to destination event e using ddCRP, where e ∈ events
        Event-Evolution-Step:
                for each e ∈ events perform event evolution
    end while
end procedure
```

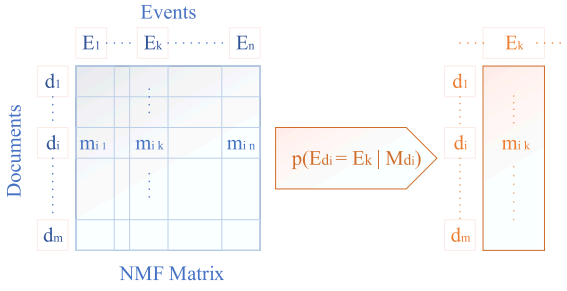


Fig. 3. Assigning documents to their specified events in the preliminary event detection component.

Algorithm 2 The preliminary event detection

```
procedure PreliminaryEventDetection
    Inputs: D ← Documents corpus of the microblog (Twitter)
    Output: Initial Events
    weightedMatrix = PreProcess(D)
    H, Z = NMF(weightedMatrix)
    for each document ∈ H do
        utilize p(Event_di = Ek | M_di) to assign document to event
    end for
    initializeEvents = individual groups of documents
    return initializeEvents
end procedure
```

and its neighbors will be assigned to the events with greater similarity. We further describe these two phases: event detection and event evolvement.

Event detection. This phase utilizes an incremental clustering framework to assign new documents to the existing events which were provided from the preliminary event detection component. The dd-CRP incremental clustering algorithm (Blei & Frazier, 2011) is used to perform the event detection process. The dd-CRP algorithm is based on the Dirichlet Process Mixture Model (Gershman & Blei, 2012) and is able to generate an infinite number of clusters. Fig. 4 illustrates how the dd-CRP algorithm works. An overview of the Bayesian network structure used in the proposed method is displayed in Fig. 5. According to this figure, a prior Dirichlet distribution is used to capture an unknown number of events. Documents are clustered according to the events which were considered as latent factors. Also, the distribution of clusters is followed by function $\theta(E_k)$. In other words, each document d_i is given via an event E_k whose significance is calculated using $\theta(E_k)$. Parameter α in the dd-CRP algorithm is used to control the event distribution, which has calculated here using the average similarity of the existing documents (Rahimpour Cami et al., 2017). The following probability function is used to determine the destination event of each new document d^*:

$$p(E_{d^*} = k|\alpha, Sim) = \begin{cases} Sim(d^*, d_{-i}), & k \in E_{-k} \\ \alpha, & k \text{ is new event} \end{cases} \quad (2)$$

where, $E_{d^*} = k$ denotes the relevance of document d^* to the kth event. Also, the $Sim()$ function calculates the similarity between document d^* and the other documents. In this paper, we utilize cosine similarity (Hasan et al., 2019) to calculate the similarity between documents.

As explained earlier, function $\theta(E_k)$ calculates the significance of event E_k in addition to representing the event distribution. This function reflects the density and freshness of an event, which is formulated in Eq. (3).

$$\theta(E_k) = \frac{\sum_{d_i \in R} \delta(d_i \in E_k) \times e^{-Age(d_i).\beta}}{N}. \quad (3)$$

As presented in Eq. (3), the parameter β is independent of the dataset and is calculated empirically. The $Age()$ function also calculates the temporal distance between the document timestamp and the current time. The parameter β represents the documents decay rate. According

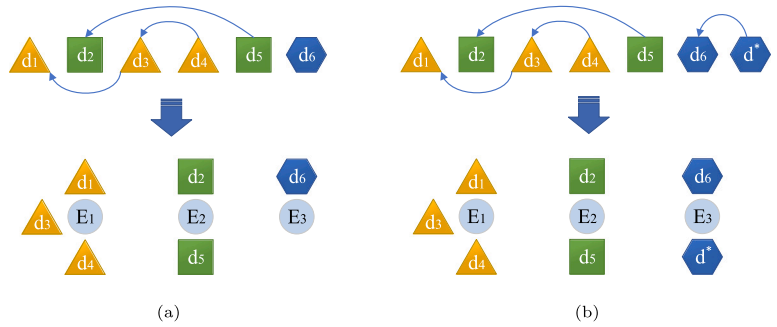


Fig. 4. Process of assigning a new document using dd-CRP: (a) Initial events (b) Assignment for adding d^*.

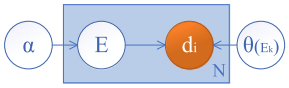


Fig. 5. Graphical view of DPMM that is used in the event evolvement tracking component.

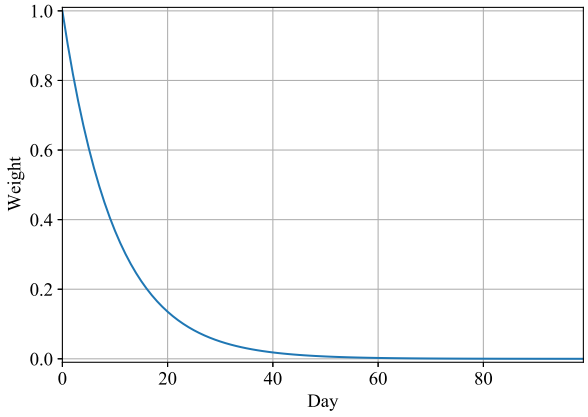


Fig. 6. Beta parameter tuning.

to the streaming behavior of social data, we empirically incorporate an exponential function ($\beta = 0.1$) which more recent documents (earlier than two weeks) should carry higher weight than the older ones. Fig. 6, demonstrates the behavior of the time functions, where β is set to be 0.1.

According to Eq. (2), the assignment of new documents is performed using dd-CRP clustering. In other words, a new document may be assigned to the existing events, or a new one may be generated, depending on its similarity to the others (Rahimpour Cami et al., 2017).

Event evolvement. The evolvement phase utilizes dd-CRP to provide a framework for reducing the noise content of events via re-assigning their constituent documents. The candidate documents are selected based on the neighborhood of the new document. The threshold parameter γ is used to select neighbors for content migration. The higher values, result to a smaller neighborhood collection and a limited content migration. In the evaluation process, we rewrite Eq. (2) via replacing d_{-i} with the neighbors of the new document. In other words, we perform dd-CRP on the neighbors of the new document. The event evolvement process is shown in Algorithm 3.

The process of event evolvement tracking component is depicted in Fig. 7. Fig. 7a displays the preliminary events. When the new document d_5 arrives, the event detection phase assigns it to event E_2 (Fig. 7b). Then, event evolvement phase selects document d_4 as a neighbor of d_5 and reassigns it to event E_2 (Fig. 7c).

Algorithm 3 Evolving events

procedure EVOLUTION
  Inputs: *initialEvents* ← Initial events set
    α, γ parameters
  Output: Updated Events
  while true **do**
    d^* = GetNewDocument()
    Event-Detection-Step:
      Sim = calculate **CosineSimilarity** (d^*, d_{-i})
      use **ddCRP**(Sim, α) for assign d^* to it destination event
    Event-Evolution-Step:
      $neighbors$ = GetNeighbors(γ)
      UpdateEvents($neighbors$)
  end while
end procedure

4. Experimental results

In this section, we describe the extensive experiments performed to evaluate the proposed method. We compute the evaluation metrics and compare the results with some of the state-of-the-art methods. Indeed, we further describe the baseline methods, the dataset and implementation framework, evaluation metrics, and experiments. Then, we analyze performance of the proposed method on the datasets. In these experiments we run the implementations 50 times and calculate the average values of metrics.

4.1. Baseline methods and toolkits

We compared the proposed method with eight methods which were selected from three aforementioned approaches (Section 2). We select MABED (Guille & Favre, 2015), NVDM (Miao et al., 2016), and GSM (Miao et al., 2017) as the feature-based approach. The metrics for MABED were calculated via public implementation[5] of this method. In NVDM and GSM, we used the reported metrics which are brought from original references. From topic-modeling approach, OLDA (Hoffman et al., 2010), SMH (Fuentes-Pineda & Meza-Ruiz, 2019), ATM (Wang et al., 2019), and ProdLDA (Srivastava & Sutton, 2017) are selected which, for OLDA the metrics are calculated via its public implementation[6] and for SMH, ATM, and ProdLDA, we used the reported metrics which were imported from original references. Also, the FSD (Petrović et al., 2010) is selected from the incremental approach for which the metrics are calculated via its public implementation[7] of this method. The hyper parameters of the above methods are adjusted using the best results.

[5] https://github.com/AdrienGuille/pyMABED.
[6] https://github.com/blei-lab/onlineldavb.
[7] https://github.com/SwatsonCodes/TwitterFirstStoryDetection.

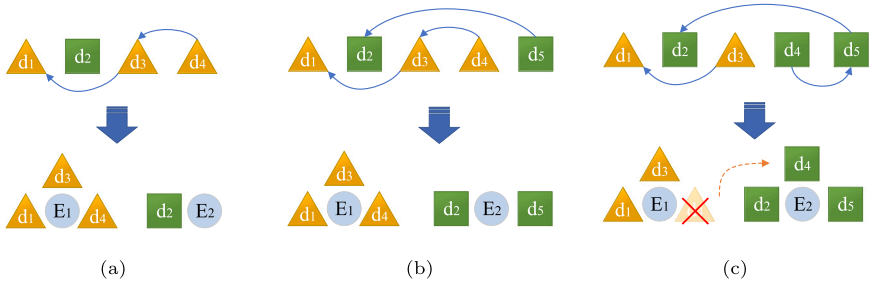


Fig. 7. Process of the event evolvement tracking component. (a) The preliminary events. (b) A new document (d_5) is arrived and it is assigned to the destination event (E_2). (c) Document d_4 migrates from the first event (E_1) to another event (E_2) due to the greater similarity with d_5 (event evolvement phase).

Table 2
The statistics of datasets.

Dataset	#Documents	#Events	#Words/Document
TwitterBreakingNews	43 M	5234	15.49
20NewsGroup	18 846	20	137.85
Grolier	30 991	–	152.56

4.2. Dataset and implementation framework

We utilize three textual datasets include TwitterBreakingNews, 20NewsGroup, and Grolier to evaluate the proposed method. In the following, a brief description of these datasets are brought. Also, the statistical information of the datasets is shown in Table 2.

TwitterBreakingNews. This dataset[8] (Kalyanam et al., 2016) was collected from Twitter social network and consists of 43 million news tweets that were collected from popular Twitter news channels, such as CNN, BBCNews, and AP, during 2013–2014. Also, tweets are tagged with 5234 events. For example, the distribution of the collected tweets (from 15 November 2013) is depicted in Fig. 8.

20NewsGroup. This dataset[9] contains 18 846 textual data which are grouped in 20 categories such as graphics and electronics. It has been used as a classical dataset in evaluating text clustering.

Grolier. This dataset[10] is built from the Grolier Encyclopedia[11] and contains about 30,000 text documents. This dataset includes information from a variety of fields such as sports, economics, and politics.

We implemented the proposed method using Python. The experiments were performed using an Intel Core i7-6700HQ CPU at 2.6 GHz with 12 GB of memory.

4.3. Evaluation metrics

We used four evaluation metric, including Accuracy (precision, recall, F1-measure) (Chen et al., 2018; Powers, 2011; Srijith et al., 2017), Entropy (Manning et al., 2008), Purity (Manning et al., 2008), and Normalized Point Mutual Information (NPMI) (Lau et al., 2014) to evaluate the performance of the proposed method, which are described below:

Accuracy: Precision/recall/F1-measure. The precision shows what percentage of detected events is the same as the reference events. The recall shows the percentage of reference events which are detected (Powers, 2011). Also, F1-measure is the average precision and recall. In this paper, we use Eqs. (4) ,(5), and (6) to calculate the precision, recall, and F1-measure, respectively.

$$Precision = \frac{|G \bigcap D|}{|D|}, \tag{4}$$

$$Recall = \frac{|G \bigcap D|}{|G|} \tag{5}$$

where G is the set of events in the ground truth dataset and D is the set of events detected. $|G \bigcap D|$ calculates the number of common words between detected event and the reference event.

$$F1 - measure = 2 \times \frac{Precision \times Recall}{Precision + Recall}. \tag{6}$$

Entropy/purity. These metrics are used to evaluate the quality of clustering. The labeled data are used to calculate these two metrics. Eqs. (7) and (8) calculate the entropy and purity, respectively.

$$\begin{aligned} Entropy_{Cluster_i} &= -\sum_{j \in Cluster_i} p(j_i) \log_2 p(j_i) \\ Total\,Entropy &= \sum_{i \in Clusters} Entropy_i \times R(i), \end{aligned} \tag{7}$$

$$Purity = \frac{1}{N} \sum_{i=1}^{k} \max_j |c_i \bigcap t_j|, \tag{8}$$

where, in Eq. (7), $R(i)$ is the ratio of the documents in an event E_i to the total available data, and in Eq. (8), N is the total number of data, k denotes the number of available clusters, c_i is a member of the cluster, and t_j is the existing labels. The output value of Eq. (8) represents the maximum number of data with the same label.

NPMI.[12] This is one of the well-known metrics in clustering which calculates the coherency of clusters. We use the NPMI metric (Zhang et al., 2020) to evaluate the coherency of detected events. We calculate the NPMI metric for a detected event E as follow:

$$NPMI(E) = \sum_{j=2}^{K} \sum_{i=1}^{j-1} \frac{\log \frac{P(w_i, w_j)}{P(w_i)p(w_j)}}{-\log P(w_i, w_j)} \tag{9}$$

where, K indicates the top-K words of event E.

4.4. Experiments

In this section, we describe several experiments which were performed to evaluate the proposed method. In the first experiment, the evolutionary behavior of the proposed method is analyzed. Also, a comparison of the performance of the proposed method compared to the mentioned methods (Section 4.1) is discussed in the next experiment.

[8] https://users.dcc.uchile.cl/~mquezada/breakingnews.
[9] http://qwone.com/~jason/20Newsgroups.
[10] https://cs.nyu.edu/~roweis/data.
[11] https://en.wikipedia.org/wiki/Grolier.

[12] We used the implementation provided by the authors in https://github.com/jhlau/topic_interpretability.

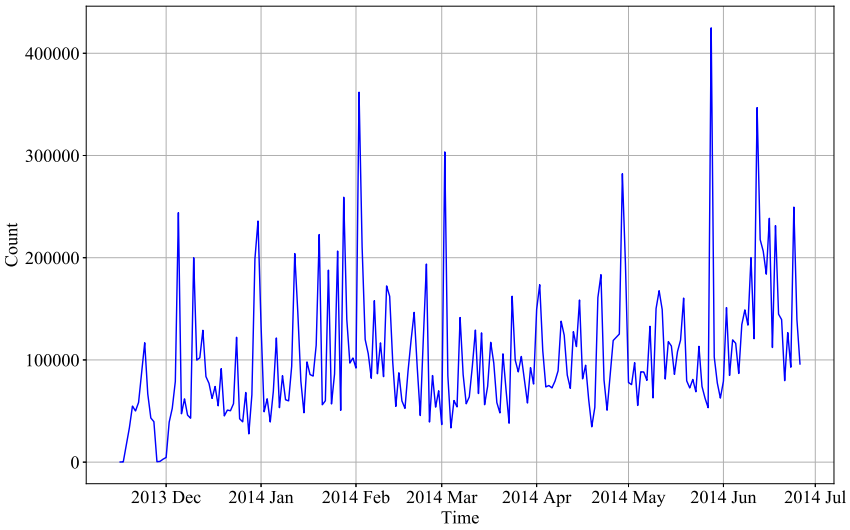


Fig. 8. Daily distribution of the number of tweets in the dataset.

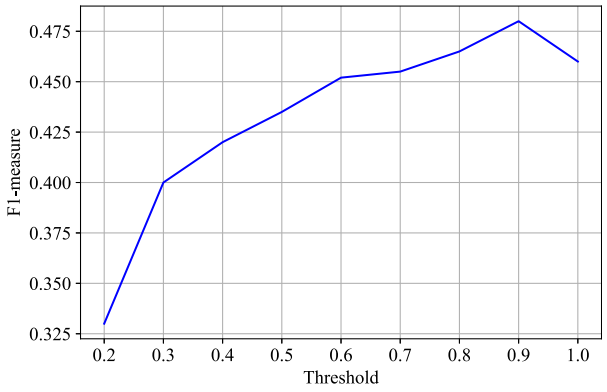


Fig. 9. Gamma parameter tuning.

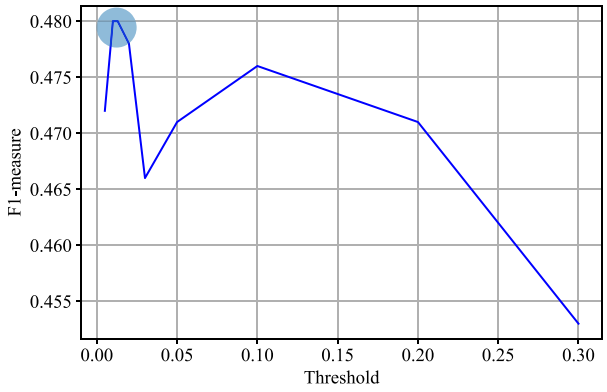


Fig. 10. Alpha parameter tuning.

In the following experiments, the concentration parameter α (in the dd-CRP algorithm) is set to the average similarity value of documents (Rahimpour Cami et al., 2017). Also, the neighborhood threshold parameter γ is calculated empirically. According to the empirical results, we experiment with γ values between 0.7 to 0.9. For example we employed 20NewsGroup dataset to conduct experiments with different γ, and α values ranging from 0 to 1 for which the results are depicted in Figs. 9 and 10. According to Fig. 9, γ values ranging from 0.7 to 0.9 yield the highest F1-measure. Also, Fig. 10 indicates average similarity of documents for α yielding the highest F1-measure.

Experiment 1. To evaluate the evolutionary behavior of the proposed method, we implemented the proposed method in two approaches including incremental and evolutionary. The aim of this experiment is to show that the evolutionary approach can perform event detection better than the incremental approach.

We employ the TwitterBreakingNews and 20NewsGroup datasets in this experiment. We ran both of them and calculated their metrics separately, with the results presented in Tables 3 and 4. Table 3 displays the results of precision, recall, and F1-measure in both incremental and evolutionary approaches. Also, Table 4 provides the results of the entropy and average purity metric of events for both the incremental and evolutionary approaches.

The graphical view of purity comparison is depicted in Figs. 11 and 12 which, indicate the results of the purity metric of the first 95 detected events in incremental and evolutionary approaches, respectively. For simplicity we only brought the results on TwitterBreakingNews dataset in these figures. The green columns represent the total number of documents in an event. The blue columns also represent the number of tweets related to the detected event.

As stated above, the aim of this experiment is to verify the evolutionary approach and to show it outperforms to the incremental approach. The results confirm that the evaluation metrics (accuracy, purity and entropy) in the evolutionary approach are better than those in the incremental approach. Superiority of the evaluation metrics in the evolutionary approach can be due to the increase in the likelihood

Table 3
Evaluation results of precision, recall, and F1-measure metrics in incremental and evolutionary approaches.

Dataset	Approach	Precision	Recall	F1
TwitterBreakingNews	Incremental	0.40	0.53	0.46
	Evolutionary	0.56	0.70	0.62
20NewsGroup	Incremental	0.39	0.51	0.44
	Evolutionary	0.41	0.58	0.48

Table 4
Evaluation result of the entropy and purity of the proposed method.

Dataset	Approach	Average Purity	Entropy
TwitterBreakingNews	Incremental	0.41	0.52
	Evolutionary	0.58	0.50
20NewsGroup	Incremental	0.65	0.39
	Evolutionary	0.66	0.38

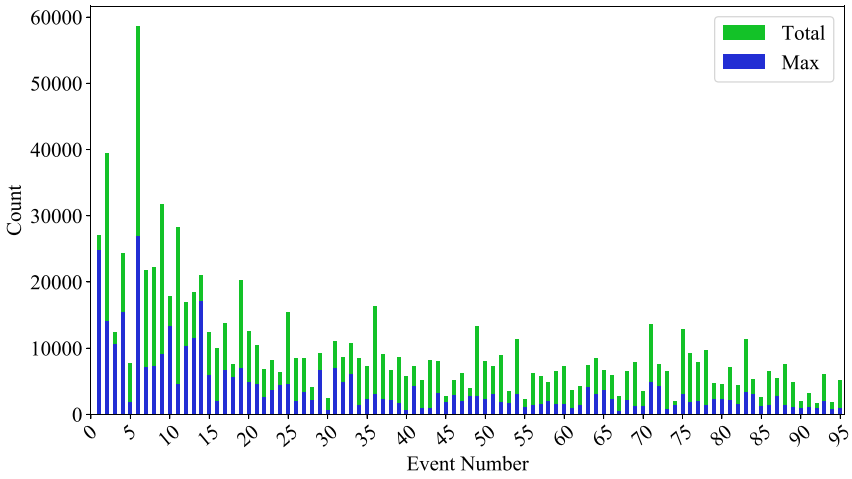


Fig. 11. Evaluation result of the purity of events in the incremental approach on TwitterBreakingNews dataset.

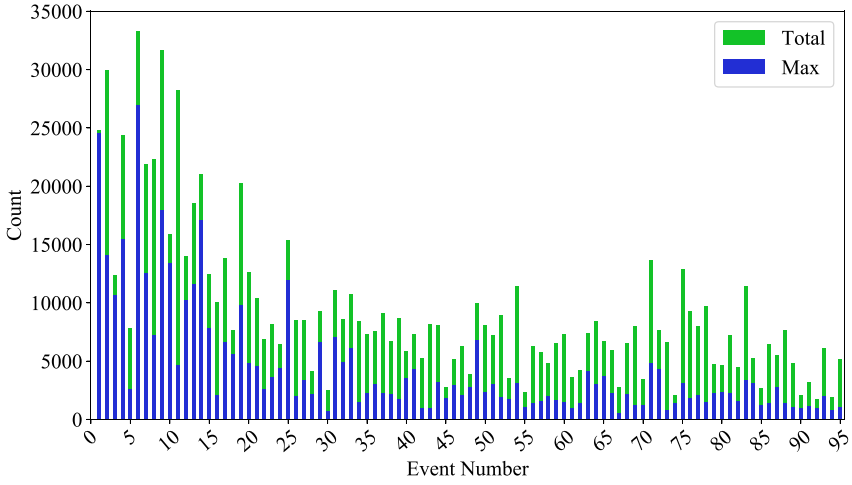


Fig. 12. Evaluation result of purity rate in the evolutionary approach on TwitterBreakingNews dataset.

of more accurate assignment of documents during the evolvement process. In other words, the evolutionary approach, not only extract events, but also captures the dynamicity of events. Therefore, it is able to provide pure events over time and improves the event detection process.

Experiment 2. The aim of this experiment is to compare the performance of the proposed method with comparing methods. According to the available data from competitors, we employ accuracy and NMPI metrics in the following two experiments. At first, we use the accuracy metrics (precision, recall, and F1-measure) and TwitterBreakingNews dataset to compare the proposed method with the three baseline methods MABED, OLDA, and FSD. Second, 20NewsGroup and Grolier dataset are used to calculate the NPMI metric for SMH, GSM, NVDM, ProdLDA, and ATM competitors.

•**Accuracy:** To calculate the accuracy metrics, we need to compare the extracted events with the reference events. To this end, the keywords of the extracted events have been compared with the keywords of the reference events. To determine whether the extracted events are true positive or false positive, we defined a similarity threshold.

In this experiment, we selected the similarity threshold to be within the range of 0.1 to 0.8. For example, we consider $E_i = \{Washington, USA, City, King\}$ as ith detected event and $G_j = \{Washington, Live, March, King\}$ as jth reference event, then, if we set the similarity threshold equal to 0.5, event E_i will be a true positive case. On the other hand, for similarity threshold equal to 0.8, event E_i will be a false positive case.

Table 5
Evaluation results of NPMI metric compared to other methods.

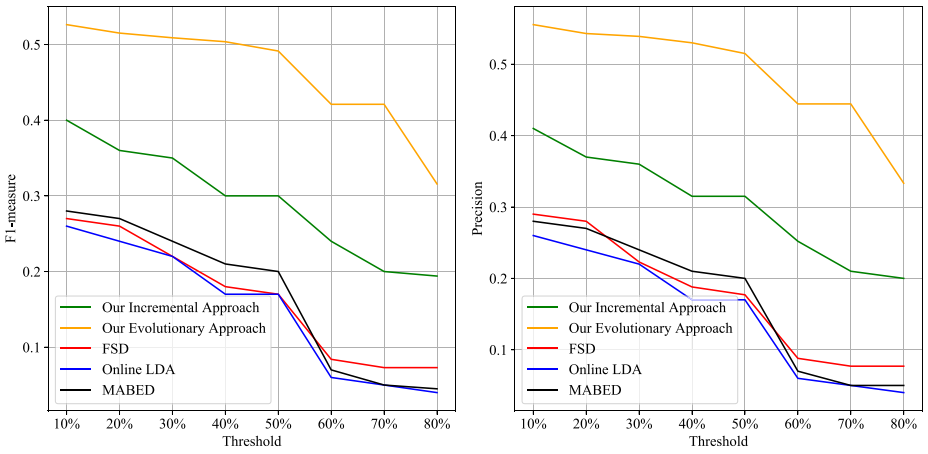
Dataset	Method					
	SMH	GSM	NVDM	ProdLDA	ATM	EEDM
20NewsGroup	0.13	0.22	0.18	0.24	[a]	**0.26**
Grolier	[a]	[a]	[a]	[a]	0.05	**0.13**

[a]Indicates that this value is not available for the correspondents.

We ran the implementation of the baseline methods (MABED, OLDA, and FSD), the proposed method without evolutionary approach, and the proposed method with evolutionary approach to calculate the precision, recall, and F1-measure for each of them. The evaluation results are shown in Fig. 13.

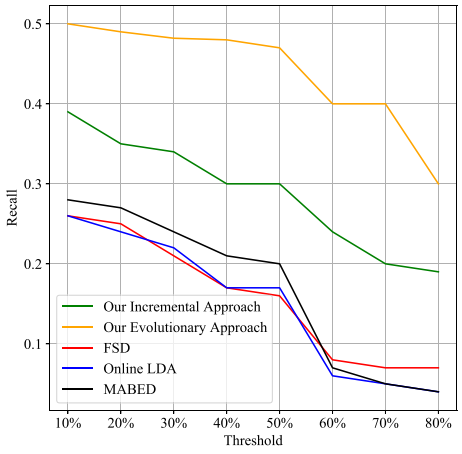
•**NPMI:** To evaluate the performance of proposed method with some state-of-the-art methods, we calculate the NPMI metric for our proposed method and compare it with SMH, ATM, GSM, NVDM, and ProdLDA. We select value of NPMI metric of SMH, ATM, GSM, NVDM, and ProdLDA methods from their original paper. Finally, we calculate the NPMI metric for the proposed method. We select detected events with top-K length (words) and employ Eq. (9) to calculate NPMI metric for each event. Table 5 shows the results of the event coherence assessment on the 20NewsGroup and Grolier datasets.

According to the experimental results, the proposed method has outperformed the existing methods (such as MABED and SMH) which



(a) F1-measure

(b) Precision



(c) Recall

Fig. 13. The results of comparing the proposed method with other existing models (a) F1-measure results (b) precision results (c) recall results.

assume the number of events is fixed. Also, due to identifying new events and evolving the existing events over time, the proposed method outperforms the incremental approach (such as FSD) in event detection.

The experimental results and comparisons with several recently developed approaches show that the proposed method outperformed the existing methods. In the first experiment, the proposed method was compared in both incremental and evolutionary approaches. The results show that the evolutionary approach has a better performance. Also, from the second experiment, results of evaluation metrics indicate the superiority of the proposed approach due to identifying new events and their evolution over time. In overall, according to the experiments, capturing the dynamicity of events and their evolvement over time (handling the adaptivity of events) improves the event detection process.

5. Conclusion

In this paper, we proposed an evolutionary model for event detection. The proposed method consisted of two main components, namely preliminary event detection and event evolvement tracking. The first component used matrix decomposition and a probabilistic function to extract the preliminary events from the initial data. A probabilistic function was applied to the output of matrix decomposition for extracting initial events. The second component used an extension of Dirichlet Process Mixture Model to assign new data and capture the dynamic behavior of events. The experimental results showed that the proposed method outperformed the existing methods. According to these results, the proposed method would improve the performance of the event detection via incorporating the evolvement approach.

For future research, in addition to the event evolvement capturing, the event prediction can also be added to the proposed method. Achieving a predictable evolutionary event detection model can improve recommender systems, crisis management, and other systems that use event detection. Also, the proposed method provides a framework, which can be used for more general feature of social data (multi-modal data).

CRediT authorship contribution statement

P.M.A. Yashar Erfanian: Methodology, Investigation, Visualization, Software, Data curation. **Bagher Rahimpour Cami:** Writing – original draft, Methodology, Validation. **Hamid Hassanpour:** Supervision, Conceptualization, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Ahmed, A., & Xing, E. (2008). Dynamic non-parametric mixture models and the recurrent chinese restaurant process: with applications to evolutionary clustering. In *Proceedings of the 2008 SIAM international conference on data mining* (pp. 219–230). SIAM.

Alsaedi, N., & Burnap, P. (2015). Arabic event detection in social media. In *International conference on intelligent text processing and computational linguistics* (pp. 384–401). Springer.

Bandaragoda, T. R., De Silva, D., & Alahakoon, D. (2017). Automatic event detection in microblogs using incremental machine learning. *Journal of the Association for Information Science and Technology*, *68*(10), 2394–2411.

Blei, D. M., & Frazier, P. I. (2011). Distance dependent Chinese restaurant processes. *Journal of Machine Learning Research*, *12*(Aug), 2461–2488.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*(Jan), 993–1022.

Cai, H., Yang, Y., Li, X., & Huang, Z. (2015). What are popular: exploring twitter features for event detection, tracking and visualization. In *Proceedings of the 23rd ACM international conference on multimedia* (pp. 89–98). ACM.

Capdevila, J., Cerquides, J., Nin, J., & Torres, J. (2017). Tweet-SCAN: An event discovery technique for geo-located tweets. *Pattern Recognition Letters*, *93*, 58–68.

Capdevila, J., Cerquides, J., & Torres, J. (2018). Mining urban events from the tweet stream through a probabilistic mixture model. *Data Mining and Knowledge Discovery*, *32*(3), 764–786.

Chen, Y., Zhang, H., Liu, R., Ye, Z., & Lin, J. (2019). Experimental explorations on short text topic mining between LDA and NMF based Schemes. *Knowledge-Based Systems*, *163*, 1–13.

Chen, X., Zhou, X., Sellis, T., & Li, X. (2018). Social event detection with retweeting behavior correlation. *Expert Systems with Applications*, *114*, 516–523.

Chowdhury, G. G. (2010). *Introduction to modern information retrieval*. Facet Publishing.

Cordeiro, M., & Gama, J. (2016). Online social networks event detection: a survey. In *Solving large scale learning tasks. Challenges and algorithms* (pp. 1–41). Springer.

Cui, W., Wang, P., Du, Y., Chen, X., Guo, D., Li, J., & Zhou, Y. (2017). An algorithm for event detection based on social media data. *Neurocomputing*, *254*, 53–58.

De Silva, D., & Alahakoon, D. (2010). Incremental knowledge acquisition and self learning from text. In *The 2010 international joint conference on neural networks (IJCNN)* (pp. 1–8). IEEE.

Diao, Q., Jiang, J., Zhu, F., & Lim, E.-P. (2012). Finding bursty topics from microblogs. In *Proceedings of the 50th annual meeting of the association for computational linguistics: Long papers-volume 1* (pp. 536–544). Association for Computational Linguistics.

Dong, X., Mavroeidis, D., Calabrese, F., & Frossard, P. (2015). Multiscale event detection in social media. *Data Mining and Knowledge Discovery*, *29*(5), 1374–1405.

Dou, W., Wang, X., Ribarsky, W., & Zhou, M. (2012). Event detection in social media data. In *IEEE visweek workshop on interactive visual text analytics-task driven analytics of social media content* (pp. 971–980). IEEE.

Fuentes-Pineda, G., & Meza-Ruiz, I. V. (2019). Topic discovery in massive text corpora based on min-hashing. *Expert Systems with Applications*, *136*, 62–72.

Gaglio, S., Re, G. L., & Morana, M. (2016). A framework for real-time Twitter data analysis. *Computer Communications*, *73*, 236–242.

Gershman, S. J., & Blei, D. M. (2012). A tutorial on Bayesian nonparametric models. *Journal of Mathematical Psychology*, *56*(1), 1–12.

Goswami, A., & Kumar, A. (2016). A survey of event detection techniques in online social networks. *Social Network Analysis and Mining*, *6*(1), 107.

Guille, A., & Favre, C. (2015). Event detection, tracking, and visualization in twitter: a mention-anomaly-based approach. *Social Network Analysis and Mining*, *5*(1), 18.

Hasan, M., Orgun, M. A., & Schwitter, R. (2019). Real-time event detection from the Twitter data stream using the TwitterNews+ Framework. *Information Processing & Management*, *56*(3), 1146–1165.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Hoffman, M., Bach, F. R., & Blei, D. M. (2010). Online learning for latent dirichlet allocation. In *Advances in neural information processing systems* (pp. 856–864).

Hofmann, T. (1999). Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval* (pp. 50–57). ACM.

Hu, L., Zhang, B., Hou, L., & Li, J. (2017). Adaptive online event detection in news streams. *Knowledge-Based Systems*, *138*, 105–112.

Huang, G., He, J., Zhang, Y., Zhou, W., Liu, H., Zhang, P., Ding, Z., You, Y., & Cao, J. (2015). Mining streams of short text for analysis of world-wide event evolutions. *World Wide Web*, *18*(5), 1201–1217.

Ibrahim, R., Elbagoury, A., Kamel, M. S., & Karray, F. (2018). Tools and approaches for topic detection from Twitter streams: survey. *Knowledge and Information Systems*, *54*(3), 511–539.

Indyk, P., & Motwani, R. (1998). Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on theory of computing* (pp. 604–613). ACM.

Kalyanam, J., Quezada, M., Poblete, B., & Lanckriet, G. (2016). Prediction and characterization of high-activity events in social media triggered by real-world news. *PLoS One*, *11*(12), Article e0166694.

Lau, J. H., Newman, D., & Baldwin, T. (2014). Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of the 14th conference of the european chapter of the association for computational linguistics* (pp. 530–539).

Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, *401*(6755), 788–791.

Lin, J., Keogh, E., Wei, L., & Lonardi, S. (2007). Experiencing SAX: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, *15*(2), 107–144.

Long, R., Wang, H., Chen, Y., Jin, O., & Yu, Y. (2011). Towards effective event detection, tracking and summarization on microblog data. In *International conference on web-age information management* (pp. 652–663). Springer.

Lu, Z., Tan, H., & Li, W. (2019). An evolutionary context-aware sequential model for topic evolution of text stream. *Information Sciences*, *473*, 166–177.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.

Miao, Y., Grefenstette, E., & Blunsom, P. (2017). Discovering discrete latent topics with neural variational inference. In *Proceedings of the 34th international conference on machine learning-volume 70* (pp. 2410–2419).

Miao, Y., Yu, L., & Blunsom, P. (2016). Neural variational inference for text processing. In *Proceedings of the 33rd international conference on international conference on machine learning-volume 48* (pp. 1727–1736).

Nguyen, D. T., & Jung, J. J. (2015). Real-time event detection on social data stream. *Mobile Networks and Applications*, *20*(4), 475–486.

Nguyen, D. T., & Jung, J. E. (2017). Real-time event detection for online behavioral analysis of big social data. *Future Generation Computer Systems*, *66*, 137–145.

Petrović, S., Osborne, M., & 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* (pp. 181–189). Association for Computational Linguistics.

Pohl, D., Bouchachia, A., & Hellwagner, H. (2015). Social media for crisis management: clustering approaches for sub-event detection. *Multimedia Tools and Applications*, *74*(11), 3901–3932.

Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *International Journal of Machine Learning Technologies*, *2*(1), 37–63.

Qian, S., Zhang, T., Xu, C., & Shao, J. (2015). Multi-modal event topic model for social event analysis. *IEEE Transactions on Multimedia*, *18*(2), 233–246.

Rahimpour Cami, B., Hassanpour, H., & Mashayekhi, H. (2017). User trends modeling for a content-based recommender system. *Expert Systems with Applications*, *87*, 209–219.

Sakaki, T., Okazaki, M., & Matsuo, Y. (2010). Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on world wide web* (pp. 851–860). ACM.

Shi, L.-L., Liu, L., Wu, Y., Jiang, L., & Hardy, J. (2017). Event detection and user interest discovering in social media data streams. *IEEE Access*, *5*, 20953–20964.

Srijith, P., Hepple, M., Bontcheva, K., & Preotiuc-Pietro, D. (2017). Sub-story detection in Twitter with hierarchical Dirichlet processes. *Information Processing & Management*, *53*(4), 989–1003.

Srivastava, A., & Sutton, C. (2017). Autoencoding variational inference for topic models. arXiv preprint arXiv:1703.01488.

Stilo, G., & Velardi, P. (2016). Efficient temporal mining of micro-blog texts and its application to event discovery. *Data Mining and Knowledge Discovery*, *30*(2), 372–402.

Unankard, S., Li, X., & Sharaf, M. A. (2015). Emerging event detection in social networks with location sensitivity. *World Wide Web*, *18*(5), 1393–1417.

Unankard, S., Li, X., Sharaf, M., Zhong, J., & Li, X. (2014). Predicting elections from social networks based on sub-event detection and sentiment analysis. In *International conference on web information systems engineering* (pp. 1–16). Springer.

Wang, R., Zhou, D., & He, Y. (2019). Atm: Adversarial-neural topic model. *Information Processing & Management*, *56*(6), Article 102098.

Wang, X., Zhu, F., Jiang, J., & Li, S. (2013). Real time event detection in twitter. In *International conference on web-age information management* (pp. 502–513). Springer.

Xie, W., Zhu, F., Jiang, J., Lim, E.-P., & Wang, K. (2016). Topicsketch: Real-time bursty topic detection from twitter. *IEEE Transactions on Knowledge and Data Engineering*, *28*(8), 2216–2229.

Xing, C., Wang, Y., Liu, J., Huang, Y., & Ma, W.-Y. (2016). Hashtag-based sub-event discovery using mutually generative lda in twitter. In *Thirtieth AAAI conference on artificial intelligence* (pp. 2666–2672).

Yang, J., Jiang, Y.-G., Hauptmann, A. G., & Ngo, C.-W. (2007). Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the international workshop on workshop on multimedia information retrieval* (pp. 197–206).

Zhang, J., Liu, M., & Zhang, Y. (2020). Topic-informed neural approach for biomedical event extraction. *Artificial Intelligence in Medicine*, *103*, Article 101783.

Zhang, C., Zhou, G., Yuan, Q., Zhuang, H., Zheng, Y., Kaplan, L., Wang, S., & Han, J. (2016). 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). ACM.

Zhao, W. X., Jiang, J., Weng, J., He, J., Lim, E.-P., Yan, H., & Li, X. (2011). Comparing twitter and traditional media using topic models. In *European conference on information retrieval* (pp. 338–349). Springer.

Zhou, X., & Chen, L. (2014). Event detection over twitter social media streams. *The VLDB Journal*, *23*(3), 381–400.

Zuo, Y., Zhao, J., & Xu, K. (2016). Word network topic model: a simple but general solution for short and imbalanced texts. *Knowledge and Information Systems*, *48*(2), 379–398.