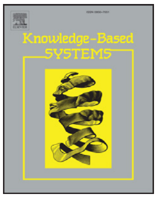
Knowledge-Based Systems 138 (2017) 105–112

Contents lists available at ScienceDirect

Knowledge-Based Systems

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



Adaptive online event detection in news streams[☆]

Linmei Hu[*], Bin Zhang, Lei Hou, Juanzi Li

Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China

ARTICLE INFO

Article history:
Received 7 June 2017
Revised 24 September 2017
Accepted 30 September 2017
Available online 9 October 2017

MSC:
00-01
99-00

Keywords:
Word embedding
Adaptive online clustering
Event detection

ABSTRACT

Event detection aims to discover news documents that report on the same event and arrange them under the same group. With the explosive growth of online news, there is a need for event detection to facilitate better navigation for users in news spaces. Existing works usually represent documents based on TF-IDF scheme and use a clustering algorithm for event detection. However, traditional TF-IDF vector representation suffers problems of high dimension and sparse semantics. In addition, with more news documents coming, IDF need to be incrementally updated. In this paper, we present a novel document representation method based on word embeddings, which reduces the dimension and alleviates the sparse semantics compared to TF-IDF, and thus improves the efficiency and accuracy. Based on the document representation, we propose an adaptive online clustering method for online news event detection, which improves both the precision and recall by using time slicing and event merging respectively. The resulted events are further improved by an adaptive post-processing step which can automatically detect noisy events and further process them. Experiments on standard and real-world datasets show that our proposed adaptive online event detection method significantly improves the performance of event detection in terms of both efficiency and accuracy compared to state-of-the-art methods.

1. Introduction

The rapidly-growing amount of electronically available information threatens to overwhelm human attention, raising new challenges for information retrieval technology. Traditional query-driven retrieval is useful for content-focused queries, but is not proper for generic queries like “What happened?” or “What's new?”. Therefore, there is a need to build automated, unsupervised methods which can simplify the operation of keeping abreast with new events that occur in the news [1–3]. Event detection is the task of discovering news documents that report on the same event and arranging them under the same group. Event detection provides a conceptual structure of the news stories and facilitates better navigation for users in news spaces.

While event detection has been studied for many years, it is still an open problem [4,5]. The most prevailing approach for event detection was proposed by Allan et al. [6] and Yang et al. [4], in which documents are processed by an online system. In such online systems, when receiving a document, the similarities between the incoming document and the known events (sometime represented by a centroid) are computed, and then a threshold is applied to determine whether the incoming document is the first story of a new event or a story of some known event. Modifications to this approach could be summarized from two aspects: better representation of contents (e.g., using named entities) [7–9] and utilizing time information [4,10]. The dominant technique for document representation in these previous work is standard TF-IDF scheme[5]. TF-IDF vector representation suffers the problems of high-dimension and semantic sparsity, leading to high computation cost and low accuracy. In addition, existing event detection methods can not automatically detect noisy events which need further processing.

In this work, we represent documents based on word embeddings and propose an adaptive online clustering algorithm for event detection. Specifically, we first learn word embeddings, and then cluster the words into different semantic classes via K-means algorithm. Words in the same cluster share the same or similar semantics. Next, we represent each news document as a distribution over the semantic classes. This representation reduces the dimension of traditional document representation (TF-IDF) and alleviates the semantic sparsity. Based on the new proposed document representation, we propose an adaptive online clustering method for event detection. Specifically, we take full advantage of the time information of news documents and arrange the news documents in chronological order. Then we divide the news documents into slices with a fixed time window size. For each time slice, we apply the single-pass online clustering method to detect events. Actually,

[☆] Fully documented templates are available in the elsarticle package on CTAN.

[*] Corresponding author.

E-mail addresses: hlm12@mails.tsinghua.edu.cn (L. Hu), zb16@mails.tsinghua.edu.cn (B. Zhang), lijuanzi@tsinghua.edu.cn (J. Li).

https://doi.org/10.1016/j.knosys.2017.09.039
0950-7051/© 2017 Elsevier B.V. All rights reserved.

this step can be done in parallel for all the time slices, which further improves the time efficiency. Since news documents reporting on the same event are usually close in time distance, time slicing ensures the precision of event detection. On the other hand, to ensure the recall, we merge events in different time slices which may refer to the same event based on similarities. Finally, we propose an adaptive method to automatically detect noisy events and further process these events. Therefore, our proposed event detection method improves event detection in terms of both efficiency and accuracy. Our contributions can be summarized as follows.

(1) We present a novel document representation method based on word embeddings, which reduces the dimension of traditional TF-IDF representation and alleviates the semantic sparsity, thus improving efficiency and accuracy of event detection.
(2) We propose a novel adaptive online clustering method, which can automatically detect noisy events and further process these events. It improves the performance of event detection in terms of both efficiency and accuracy significantly.
(3) Experiments on standard and real-world datasets show that our proposed adaptive online event detection method significantly outperforms state-of-the-art event detection methods.

The remainder of this paper is organized as follows. In Section 2, we formulate the event detection problem. In Section 3, we detail our proposed method. Section 4 describes our experimental results. In Section 5, we review the related literature, followed by conclusion and future research directions in Section 6.

2. Problem definition

In this section, we first define some concepts as well as the problem of event detection.

News stream. News documents from various sources usually form a stream in chronological order. A news stream $D = \{d_1, \ldots, d_m, \ldots\}$ is a sequence of documents. d_i is associated with a pair (d_i, t_i), where d_i is a document comprising a sequence of words and t_i is the publishing time in non-descending order, i.e. $t_i \le t_{i+1}$.

Event. An *event* is a particular thing that happened at a specific time and place [6,11], and an event is usually composed of a set of news documents reporting on it. We consider an event $E = \{d_1, \ldots, d_M\}$ as a sequence of news documents.

For example, "2010 Chile earthquake" is an event. It consists of sequential news documents describing different aspects of the event such as *rescue efforts, damages, chaos*, and so on.

Event detection. The task of event detection is to discover news documents reporting on the same event from a news stream $D = \{d_1, d_2, \ldots, d_m, \ldots\}$ and divide them into event-centric clusters $\{E\}$, where $E = \{d_1, \ldots, d_M\}$ according to the events they report on.

Event detection typically has two major components: 1) document representation and 2) cluster analysis [12]. Document representation handles with translating documents into structures appropriate for clustering. This is generally completed by representing documents numerically as vectors and matrices. Cluster analysis contains methods for designing meaningful data clusters from the data structure formed by document representation methods.

Traditional methods usually use TF-IDF vectors to represent documents, which leads to high dimension and sparse semantics. In addition, IDF needs to be incrementally updated when more and more new documents come. Therefore, we propose a new document representation based on word embeddings, which avoids the problems of TF-IDF. Based on the document representation, a lot of clustering methods such as K-means, LDA and single-pass online clustering can be applied for event detection. However, K-means and LDA need a prior knowledge of cluster number, which is quite hard to determine. The number of clusters has a big influence on the clustering results. On the other hand, it is more convenient to control the similarity threshold of single-pass online clustering. In addition, single-pass online clustering is a one-pass algorithm which is much more efficient. Therefore, we propose a new adaptive online clustering algorithm based on single-pass online clustering. Our proposed clustering algorithm further considers improving the efficiency and accuracy of online event detection via time slicing and adaptive post-processing respectively.

3. Our method

In this section, we detail our proposed adaptive online event detection method to automatically group news documents according to the events they report on. As a result, each event is composed of a sequence of news documents. We first present our document representation based on word embeddings and then describe our adaptive online clustering algorithm for event detection.

3.1. Document representation

To alleviate the problems of traditional TF-IDF representation, we propose a novel document representation based on word embeddings. As shown in Fig. 1, our proposed document representation consists of three steps: word embedding, word clustering and document vectorization.

Word Embedding. Word is the basic element in a document, so we first transform words into continuous low-dimensional vectors. Let $\mathcal{V}$ denote the vocabulary in all the news documents D, we employ skip-gram model [13] to learn a mapping function: $\mathcal{V} \rightarrow R^M$, where R^M is the M-dimensional representation of w_i. Specifically, given a document $d \in D$ associated with word sequence $w_1, w_2, \ldots, w_N$, skip-gram model maximizes the co-occurrence probability among words that appear within a contextual window k:

$$\max_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} \sum_{j=i-k}^{i+k} \log p(w_j|w_i) \tag{1}$$

The probability $p(w_j|w_i)$ is formulated as:

$$p(w_j|w_i) = \frac{\exp(\mathbf{w}_j^T \mathbf{w}_i)}{\sum_i \exp(\mathbf{w}_i^T \mathbf{w}_i)} \tag{2}$$

where $\mathbf{w}_i$ is the vector representation of word w_i. Different from TF-IDF vector representation, word embeddings do not need to be incrementally updated with more and more new documents coming since we can use a large independent news corpus to train word embeddings.

Word Clustering. Traditional TF-IDF representations suffer problems of the curse of dimensionality and feature independence assumption. These methods often ignore the semantic relationships among word features which leads to document sparse representation with many zero features values. If there are two documents describing similar events but using different words, they have difficulty in making a correct decision that they belong to the same event [7]. Thus, traditional text processing based on keyword comparison could not provide good performance. To reduce the semantic sparsity, we use K-means clustering to cluster words referring to the same or similar meaning to obtain a latent semantic space.

Document vectorization. At last, for each document, we can replace the words with corresponding word cluster indexes. Then a document can be easily represented by the distribution of word clusters. By representing the documents in the same latent space, we can alleviate the problems of high dimension and semantic

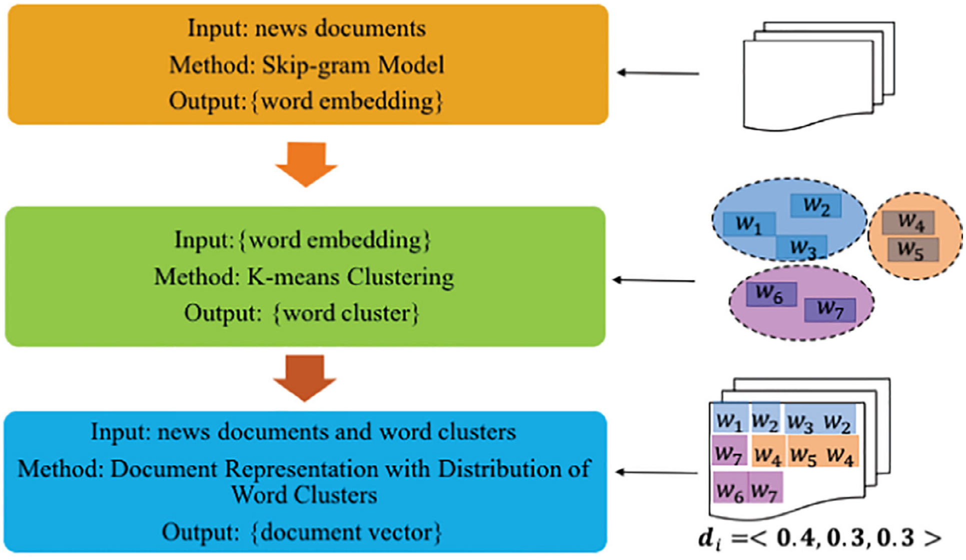

Input: news documents
Method: Skip-gram Model
Output: {word embedding}
Input: {word embedding}
Method: K-means Clustering
Output: {word cluster}
Input: news documents and word clusters
Method: Document Representation with Distribution of Word Clusters
Output: {document vector}
w_1 w_2 w_3 w_4 w_5 w_6 w_7
w_1 w_2 w_3 w_2
w_7 w_4 w_5 w_4
w_6 w_7
d_i =< 0.4, 0.3, 0.3 >

Fig. 1. Process of document representation.

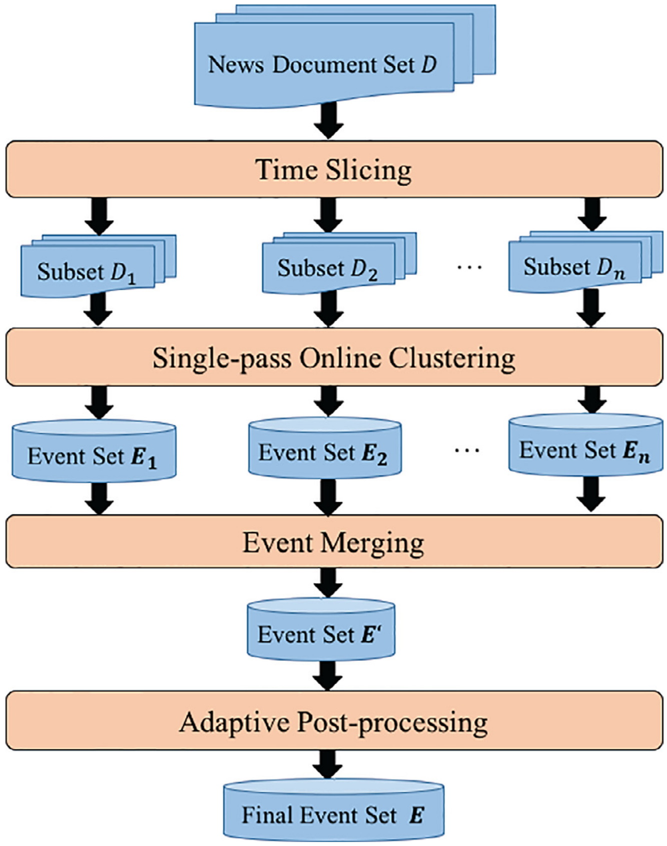


Fig. 2. Process of adaptive online clustering method.

sparsity. Thus, our proposed document representation is more efficient and accurate.

3.2. Adaptive online clustering

After document representation, we propose an adaptive online clustering method for news event detection as illustrated in Fig. 2. Our method consists of four steps: time slicing, single-pass online clustering, event merging, and adaptive post-processing. We detail the steps as follows.

Time Slicing. In event detection, time information is very important. After an event happens, a large number of relevant news documents usually come out in a short time. News documents with a longer time interval are usually about different events. Therefore, the division of news documents according to publishing time information is helpful for spotting news documents reporting on the same event. In particular, we arrange the news documents in chronological order and divide the news documents into clusters with a fixed time window size (e.g., a week). After division, we get news document clusters where each cluster corresponds to a time slice.

Single-pass online clustering. For each time slice, we apply single-pass online clustering to detect events. This step can be done in parallel for all the time slices, which further improves time efficiency. We review the basic single-pass online clustering algorithm as shown in Algorithm 1 . The single-pass online clustering algorithm is quite simple. It sequentially processes the input documents, one at a time, and grows clusters incrementally. A new document is absorbed by the most similar cluster generated previously, if the similarity (we use cosine similarity) between this document and the centroid of that cluster is above a pre-selected threshold; otherwise, the document is treated as the seed of a new cluster. Finally, the algorithm returns event-centric clusters (events) $\{E_1, E_2, \ldots, E_K\}$ in each time slice. The time complexity of single-pass online clustering is O(MK), where M is the number of news documents and K is the number of clusters.

Event merging. After detecting events in each time slice, we merge the events in different time slices but referring to the same event. In this way, we can ensure high recall of event detection. The event merging algorithm is illustrated in Algorithm 2 . We incrementally merge each time-slice events $\mathbf{E}_t = \{E_{ti}\}$ $(t = 2, \ldots, T)$ with former events $\mathbf{E}$ (initially, $\mathbf{E} = \mathbf{E}_1$). Each event E is represented by the centroid vector ($\mathbf{c} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{d}_i$) of the corresponding event-centric cluster. The event is merged with the most similar former one, if the similarity between the two events is larger than a pre-defined merging threshold Δ. Finally, the algorithm returns the event set $\mathbf{E}$=$\{E_1, E_2, \ldots, E_K\}$, where there are no events referring to the same event.

Adaptive post-processing. However, some resulted events (e.g., events about entertainment) in $\mathbf{E}$=$\{E_1, E_2, \ldots, E_K\}$ are still noisy,

```
Algorithm 1: Single-Pass Online Clustering.
Input: News documents $\mathbf{D}$={$d_1, d_2, \ldots d_M$}, Similarity
       Threshold $\delta$
Output: Event-centric clusters $\mathbf{E}$={$E_1, E_2, \ldots, E_K$}
for the $j - th$ document $d_j$ do
    calculate the vector representation $\mathbf{d}_j$ of $d_j$
    if $E = \emptyset$ then
        create $E_1$
        let $d_j \in E_1$
        represent $E_1$ by $\mathbf{d}_j$
    else
        foreach event-centric cluster $E_k$ do
            calculate the similarity $sim(d_j, E_k)$
            let $maxS = \max_j sim(d_j, E_k)$
            let $maxE = E_k | sim(d_j, E_k) = maxS$
            if $maxS \geq \delta$ then
                let $d_j \in maxE$
                recalculate the representation of $maxE$ by the
                centroid
            else
                create a new event-centric cluster $E_{new}$
                let $d_j \in E_{new}$
                represent $E_{new}$ by $\mathbf{d}_j$
    j++
return all event-centric clusters $E_1, E_2, \ldots, E_K$
```

```
Algorithm 2: Event Merging.
Input: Event sets {$\mathbf{E}_t$}, where $\mathbf{E}_t = \{E_{ti}\}$ contains a set of
       events in the $t$-th time-slice, Merging Threshold $\Delta$
Output: Event set $\mathbf{E}$={$E_1, E_2, \ldots, E_K$}
let $\mathbf{E} = \mathbf{E}_1$
foreach event set $\mathbf{E}_t$ (t=2,… ,T) do
    foreach event $E_{ti} \in \mathbf{E}_t$ do
        foreach event $E_k \in \mathbf{E}$ do
            let $c_{ti}$, $c_k$ represent the centroid of $E_{ti}$ and $E_k$
            respectively
            calculate the similarity $sim(c_{ti}, c_k)$
            let $maxS = \max_i sim(c_{ti}, c_k)$
            let $maxE = E_k | sim(c_{ti}, E_k) = maxS$
            if $maxS \geq \Delta$ then
                let $maxE \leftarrow E_{tj} \cup maxE$
                recalculate the representation of $maxE$ by the
                centroid
            else
                let $E_{ti} \in \mathbf{E}$
return all event-centric clusters $\mathbf{E} = \{E_1, E_2, \ldots E_K\}$
```

consisting of similar news documents belonging to different events. To ensure the quality of the event-centric clusters, we present an *adaptive* method to deal with the noisy clusters automatically. In particular, for each event-centric cluster, we calculate the similarities between each news document and the centroid ($sim(d, c) = \frac{\mathbf{d}\cdot\mathbf{c}}{|\mathbf{d}|\cdot|\mathbf{c}|}$), and then obtain the variance ($var = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$, where x_i stands for $sim(d_i, c)$ and $\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$) of the similarity values. The high-quality clusters typically have low variance values, while low-quality clusters have high variance values. Thus, we can reapply the basic single-pass online clustering algorithm (using higher similarity threshold value δ) on the clusters with variance value $var \geq V$. V is a predefined noisy threshold. The clusters with high quality ($var < V$) are remained. Finally, the obtained clusters are our resulted events.

In summary, the above two steps, i.e., document representation and adaptive online clustering compose our adaptive online event detection method. It improves document representation, and optimizes clustering with time slicing, event merging and adaptive post-processing.

4. Experiments

To evaluate the effectiveness of our proposed event detection method, we conducted experiments on both standard datasets and real-world datasets. We utilized the standard dataset TDT4 [14] which contains 6000 news documents in 3 languages from October 1, 2000 to January 31, 2001. Each document is annotated to belong to one event out of 100 events. We used the Chinese (TDT4-Ch) and English (TDT4-En) news documents. TDT4-Ch contains 1023 news documents covering 38 events and TDT4-En contains 1887 news documents covering 70 events respectively.

We also tested our method on a real-world Chinese news dataset: NewsMiner. We constructed the dataset from our online news analysis system NewsMiner.[1] It crawls Chinese news documents from various sources, stores and analyzes the news data [15]. We used the news documents from November 20, 2016 to January 18, 2017 in the system. In total, we collected about 600,000 news documents.

4.1. Experimental setting

We compared the performance of our proposed method with that of several state-of-the-art methods on two standard datasets (with ground-truth results) based on two kinds of metrics. In our method, for document representation, we train word embeddings (the dimension of word embedding is set to 100) using the dataset itself as we compared with the result of using an external large corpus and found no big difference. When using K-means to cluster words, we set $K = 1000$. For adaptive online clustering, we chose the parameter values which lead to the best clustering result. Thus, for TDT4-Ch dataset, we set $\delta = 0.4$, $\Delta = 0.5$, $V = 0.02$ and for TDT4-En dataset, we set $\delta = 0.3$, $\Delta = 0.7$, $V = 0.05$.

Baselines. Our baseline methods include:

1) **LDA**. LDA is a widely used topic model for text analysis [16]. After applying LDA, we can get the distribution of documents over topics. We assign each document to the topic with the largest probability. Each topic is considered as an event. We set the topic number to the true number of events (i.e., K=3870 for TDT4-Ch and TDT4-En respectively). The number of iterations is set to 1000. The prior hyperparameters α and β are empirically set to $50/K$ and 0.1 respectively.
2) ***K*-means1**. K-means clustering is a popular method for cluster analysis in data mining. We represent each document as a TF-IDF vector and set the cluster number to the true number of events (i.e., $K = 3870$ for TDT4-Ch and TDT4-En respectively).
3) ***K*-means2**. Different from ***K*-means1** method, we use the word embedding based document representation which is proposed in this paper and set the cluster number to the true number of events (i.e., $K = 3870$ for TDT4-Ch and TDT4-En respectively).
4) **Single-pass Online clustering**. We represent each document as a TF-IDF vector and apply the basic single-pass online clustering method. We choose the similarity threshold which leads to the best clustering result (i.e., $\delta = 0.15, 0.05$ for TDT4-Ch and TDT4-En respectively).
5) **Average-of-Words Embedding**. We represent each document as bag-of-words embedding (average all words' vector in the

[1] http://newsminer.net.

document). Then we use our proposed adaptive online clustering for event detection.

6) **Sum-of-Words Embedding**. We represent each document as bag-of-words embedding (sum up all words' vector in the document). Then we use our proposed adaptive online clustering for event detection.

7) **Doc2Vec**. We use Doc2Vec[2] introduced in [17] to represent the documents by embeddings. The dimension of document embedding is set as the same as that of word embedding, i.e., 100. Then we use our proposed adaptive online clustering for event detection.

Evaluation metrics. We use two kinds of metrics. We detail them as follows.

1) *F1*. Consider the two clustering systems (true and computed) as categorizations of pairs of documents: in the same cluster or in different clusters. Precision is defined as the fraction of pairs of documents that are actually in the same category out of those that are computed to be. Recall is defined as the fraction of pairs of documents that are computed to be in the same category out of those that are actually in the same category. F1-measure is the harmonic mean of precision and recall. Formally,

$$\begin{aligned} P &= \frac{TP}{TP+FP} \\ R &= \frac{TP}{TP+FN} \\ F1 &= \frac{2P \cdot R}{P+R}, \end{aligned} \tag{3}$$

where TP is the number of pars of documents which are correctly divided into the same cluster, FP is the number of pairs of documents which are incorrectly divided into the same cluster and FN is the number of pairs of documents which are incorrectly divided into different clusters.

2) *NMI*. The Normalized Mutual Information (NMI) is widely used to evaluate the quality of the clustering results. NMI measures the amount of statistical information shared by the random variables representing the cluster assignments and the groundtruth groups of the documents. Normalized Mutual Information (NMI) is formally defined as follows [18]:

$$NMI = \frac{\sum_{c,k} n_{c,k} \log(\frac{N \cdot n_{c,k}}{n_c \cdot n_k})}{\sqrt{\sum_c n_c \log \frac{n_c}{N} \cdot \sum_k n_k \log \frac{n_k}{N}}}, \tag{4}$$

where n_c is the number of documents in class *c*, n_k is the number of documents in cluster *k*, $n_{c,\,k}$ is the number of documents in class *c* as well as in cluster *k*, and *N* is the number of documents in the dataset. When the clustering results perfectly match the groundtruth classes, the NMI value will be one, while when the clustering results are randomly generated, the NMI value will be close to zero.

4.2. Experimental results

Overall Results. From Tables 1 and 2, we can see that on both standard datasets, our method outperforms state-of-the-art baselines significantly. Although the number of topics (clusters) in LDA and *K*-means methods is set to the true number of clusters, they get lower performance. The single-pass online clustering algorithm performs much better than LDA and *K*-means. *K*-means2 with our proposed document representation achieves better results than *K*-means1, which shows the proposed word embedding based document representation is better than traditional TF-IDF vector representation. The last four lines of the tables show that our proposed document representation method significantly outperforms the state-of-the-art document representation methods. Doc2Vec method performs much better than bag-of-words embedding methods (averaging or summing up all words). Averaging all words' vectors to represent a document is slightly better than summing up the vectors. Figs. 3 and 4 show comparison of all the methods on TDT4-Ch and TDT4-En respectively in terms of precision, recall and F1 values in detail. As we can see, our method gets the highest recall value and F1 value on both datasets. LDA gets the highest precision value, while it achieves lower F1 value due to its low recall. Doc2Vec method performs better than *K*-means1 but slightly worse than *K*-means2. Overall, *K*-means1, Average-of-Words Embedding, Sum-of-Words Embedding perform the worst in terms of both precision and recall.

Table 1
The clustering results of different methods on TDT4-Ch.

Methods	F1	NMI
LDA	0.765	0.898
K-means1	0.535	0.719
K-means2	0.688	0.855
Single-pass online clustering	0.785	0.903
Average-of-Words Embedding	0.572	0.794
Sum-of-Words Embedding	0.556	0.780
Doc2Vec	0.679	0.845
Our method	0.825	0.937

Table 2
The clustering results of different methods on TDT4-En.

Methods	F1	NMI
LDA	0.606	0.832
K-means1	0.529	0.757
K-means2	0.590	0.811
Single-pass online clustering	0.790	0.861
Average-of-Words Embedding	0.276	0.581
Sum-of-Words Embedding	0.263	0.539
Doc2Vec	0.557	0.796
Our method	0.872	0.883

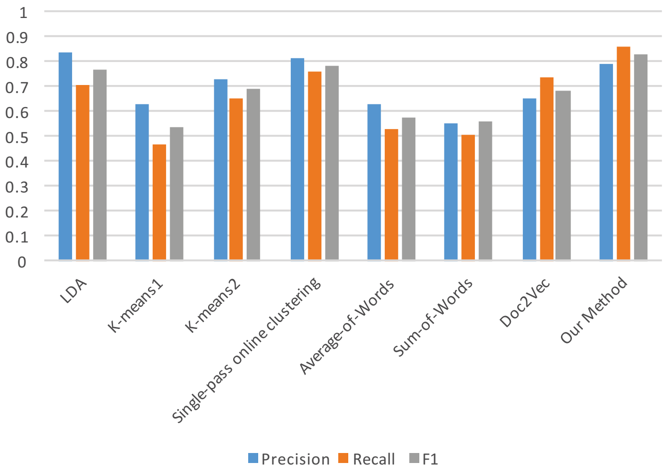


Fig. 3. Comparison of different methods in terms of precision, recall, and F1 measure on TDT4-Ch.

Influence of Similarity Threshold δ. We also investigated the influence of similarity threshold. Clustering performance of our event detection method with different values of similarity threshold δ on the two standard datasets are illustrated in Figs. 5 and 6 respectively. As we can see, on both datasets, the F1 value and NMI consistently grow with the increase of δ at first, and then reach

[2] https://radimrehurek.com/gensim/models/doc2vec.html.

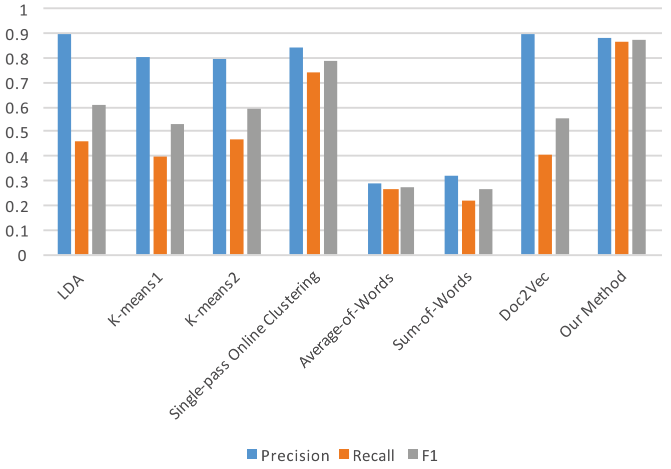


Fig. 4. Comparison of different methods in terms of precision, recall, and F1 measure on TDT4-En.

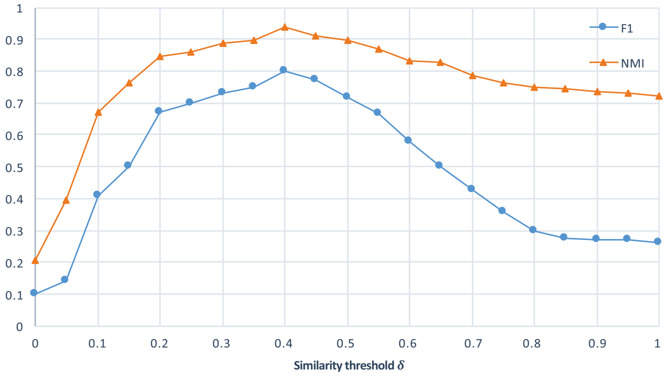


Fig. 5. Performance of our event detection method with different values of similarity threshold δ on TDT4-Ch.

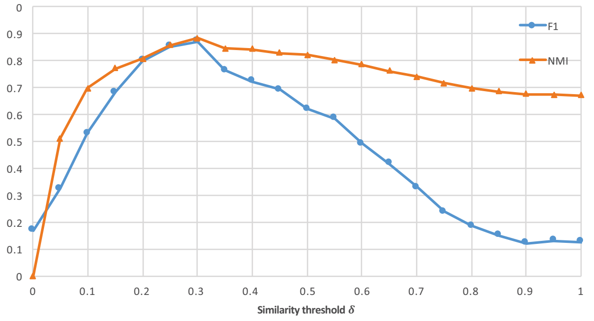


Fig. 6. Performance of our event detection method with different values of similarity threshold δ on TDT4-En.

the peak. Thereafter, they drop quickly with the further increase of δ. The reason is that at the beginning, when the similarity threshold is small, many news documents that belong to different events are clustered into the same event, resulting in low precision. Thus, the F1 value and NMI are low. With the increase of δ, the precision grows and thus, F1 and NMI grow. After F1 and NMI reach the optimal value, as the similarity threshold δ continues to increase, many news documents belonging to the same event are clustered into different events, resulting in a drop of recall, and thus, F1 and NMI decrease. In addition, we can see that for TDT4-Ch and TDT4-En, the F1-measure and NMI consistently reach the peak at $\delta = 0.4$ and $\delta = 0.3$ respectively.

4.3. Time efficiency

In online event detection, time efficiency is very important. We compare the time efficiency of our method with that of baseline methods on two standard datasets. As shown in Table 3, our method significantly outperforms state-of-the-art methods in terms of time efficiency. *K*-means1 and single-pass online clustering using TF-IDF vector representation take the longest time. *K*-means2 using the word embedding based document representation significantly improves *K*-means1 and single-pass online clustering (more than 10 times). It demonstrates the proposed document representation based on word embeddings is very efficient.

Our method uses document representation based on word embeddings, and divides news documents into subsets according to time slices, where news documents in different slices can be processed in parallel. Thus, it is much more efficient than all the baseline methods.

4.4. Visualization

We applied the proposed online event detection method on a real-world dataset named NewsMiner to verify the effectiveness of the proposed method. We set the time window to 7 days and obtain 9 time slices from November 20, 2016 to January 18, 2017. The parameter setting is the same as that on TDT4-Ch. Finally, we get about 80,000 events in total. We show the hot events along the timeline in Fig. 7. It can be seen that the hot events during the period of November 20, 2016 to January 18, 2017 mainly include “*Xi Jinping gives a talk at APEC meeting*”, “*Lin Dan’s infidelity event*”, “*Nanjing Massacre Victims National Commune Day*”, “*Xi Jinping attended the World Economic Forum*”, “*Turkey was attacked on New Year’s Eve*”, “*South Korea Park Scandal*” and so on. The obtained events well accord with the events that happened in real life, which demonstrates the effectiveness of our proposed adaptive online event detection method.

5. Related work

The objective of event detection is to identify stories in several continuous news streams that pertain to new or previously unidentified events [4]. Event detection task is unsupervised and can be divided into two forms: retrospective detection and online detection [1,4,11]. The former aims to discover previously unidentified events in a chronologically ordered accumulation of news documents (stories), and the latter strives to identify the onset of new events from live news feeds in real time. Both forms of detection intentionally lack prior knowledge of novel events but do have access to unlabeled historical news stories for use as contrast sets.

Most of the proposed event detection algorithms, retrospective or online, were developed based on the document clustering approach. Yang et al. [4,11] implemented two clustering methods for event detection: group average clustering algorithm (GAC) and incremental clustering algorithm (INCR). GAC, operating in a strict retrospective detection setting, makes use of divide and conquer strategy performing agglomerative clustering. On the other hand, INCR, designed for both retrospective and online detection, is a single-pass incremental clustering algorithm that processes the documents sequentially and produces clusters incrementally [4,11].

INCR [1,4] was the most prevailing approach, in which documents are processed by an on-line system. In such online systems, a document is absorbed by the most similar cluster in the past if the similarity between the document and the cluster is greater than a preselected clustering threshold; otherwise, the document becomes the seed of a new cluster. Modifications to this approach may be summarized from two aspects: better representation of

Table 3
Runtime of different methods on different datasets.

	LDA	*K*-means1	*K*-means2	Single-pass online clustering	Average-of-Words Embedding	Sum-of-Words Embedding	Doc2Vec	Our method
TDT4-Ch	240s	4936s	85s	1956s	6s	6s	5s	6.48s
TDT4-En	635s	5135s	470s	3150s	8s	8s	7s	9s

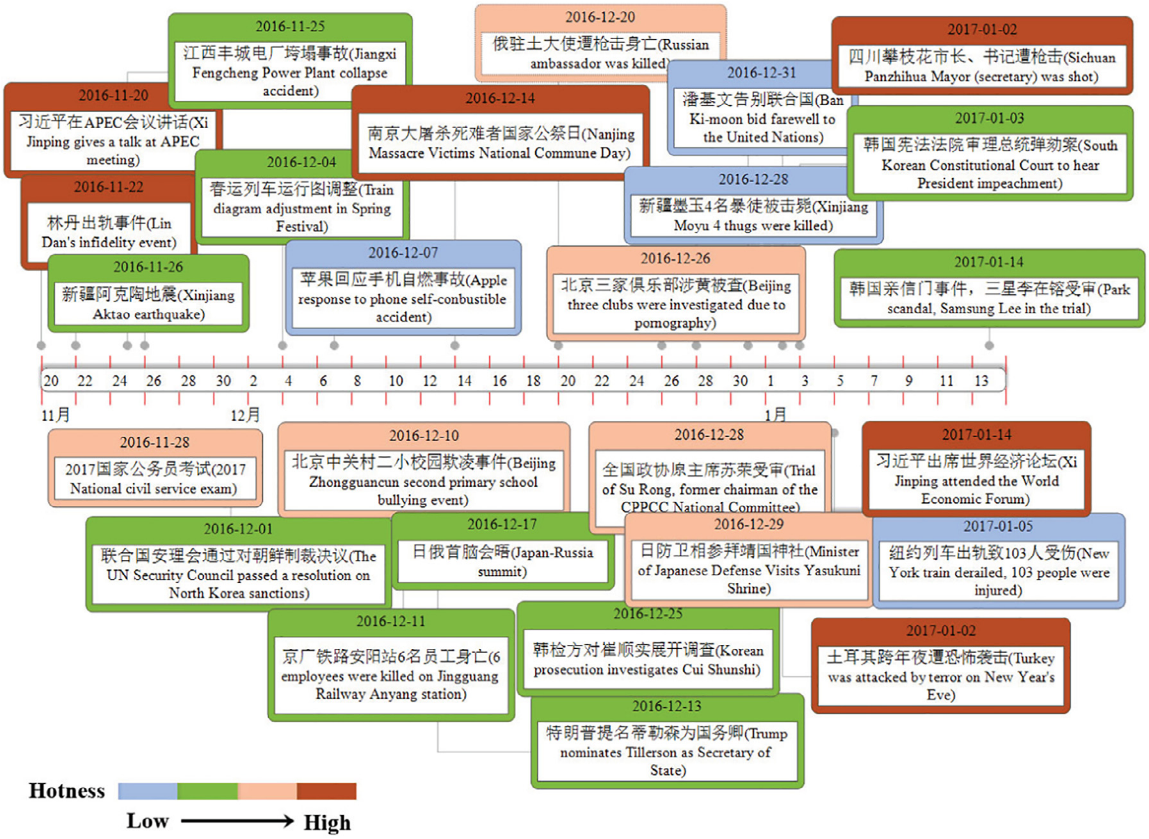

2016-11-25 江西丰城电厂垮塌事故(Jiangxi Fengcheng Power Plant collapse accident)
2016-12-20 俄驻土大使遭枪击身亡(Russian ambassador was killed)
2017-01-02 四川攀枝花市长、书记遭枪击(Sichuan Panzhihua Mayor (secretary) was shot)
2016-11-20 习近平在APEC会议讲话(Xi Jinping gives a talk at APEC meeting)
2016-12-14 南京大屠杀死难者国家公祭日(Nanjing Massacre Victims National Commune Day)
2016-12-31 潘基文告别联合国(Ban Ki-moon bid farewell to the United Nations)
2017-01-03 韩国宪法法院审理总统弹劾案(South Korean Constitutional Court to hear President impeachment)
2016-12-04 春运列车运行图调整(Train diagram adjustment in Spring Festival)
2016-11-22 林丹出轨事件(Lin Dan's infidelity event)
2016-12-28 新疆墨玉4名暴徒被击毙(Xinjiang Moyu 4 thugs were killed)
2016-11-26 新疆阿克陶地震(Xinjiang Aktao earthquake)
2016-12-07 苹果回应手机自燃事故(Apple response to phone self-conbustible accident)
2016-12-26 北京三家俱乐部涉黄被查(Beijing three clubs were investigated due to pornography)
2017-01-14 韩国亲信门事件，三星李在镕受审(Park scandal, Samsung Lee in the trial)
20 22 24 26 28 30 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 1 3 5 7 9 11 13
11月
12月
1月
2016-11-28 2017国家公务员考试(2017 National civil service exam)
2016-12-10 北京中关村二小校园欺凌事件(Beijing Zhongguancun second primary school bullying event)
2016-12-28 全国政协原主席苏荣受审(Trial of Su Rong, former chairman of the CPPCC National Committee)
2017-01-14 习近平出席世界经济论坛(Xi Jinping attended the World Economic Forum)
2016-12-01 联合国安理会通过对朝鲜制裁决议(The UN Security Council passed a resolution on North Korea sanctions)
2016-12-17 日俄首脑会晤(Japan-Russia summit)
2016-12-29 日防卫相参拜靖国神社(Minister of Japanese Defense Visits Yasukuni Shrine)
2017-01-05 纽约列车出轨致103人受伤(New York train derailed, 103 people were injured)
2016-12-11 京广铁路安阳站6名员工身亡(6 employees were killed on Jingguang Railway Anyang station)
2016-12-25 韩检方对崔顺实展开调查(Korean prosecution investigates Cui Shunshi)
2017-01-02 土耳其跨年夜遭恐怖袭击(Turkey was attacked by terror on New Year's Eve)
2016-12-13 特朗普提名蒂勒森为国务卿(Trump nominates Tillerson as Secretary of State)
Hotness
Low → High

Fig. 7. Visualization of detected hot events on NewsMiner.

contents and utilizing of time information. From the aspect of utilizing the contents, TF-IDF is still the dominant technique for document representation, and cosine similarity is the generally used similarity metric. Some works focus on finding new distance metrics, such as the Hellinger distance metric [10]. But more works focus on finding better representations of documents, i.e. feature selection. Yang et al. [8] classified documents into different categories, and then removed stop words with respect to the statistics within each category. Allan et al. [19], Yang et al. [8] and Lam et al. [7] tried to use named entities. In [8], Yang et al. proposed to re-weight both named entities and non-named terms with respect to statistics within each category. Zhang et al. [9] utilized news indexing-tree to speed up the task and proposes two term reweighing approaches to improving accuracy. A recent publication of Kumaran and Allan [20] summarized the work in this direction and proposed some extensions. They exploited to use both text classification and named entities to improve the performance of event detection. From the aspect of utilizing time information, Yang et al. [4] and Cordeiro [10] used decaying functions to modify the similarity metrics of the contents.

Li et al. [5] first proposed a probabilistic model for retrospective event detection. With the success of the topic model LDA (Latent Dirichlet Allocation) in topic analysis of text, some works used it for event detection. Keane et al. [21] leverages LDA to generate topics which are then later labeled as Eventy-Topics or non-Eventy-Topics and then calculates the cosine similarity between the topics of the daily topic models.

There are also a lot of works focused on event detection on informal texts. Graph-based clustering algorithms have been proposed for the task. Long et al. [22] used a hierarchical divisive clustering approach on a co-occurrence graph (connecting messages according to word co-occurrence) to divide topical words into event clusters. A modularity-based graph partitioning technique was used to form events by splitting the graph into subgraphs each corresponding to an event [23]. In general, hierarchical clustering algorithms do not scale to the large size of the data because they require the full similarity matrix, which contains the pairwise similarity between groups [24,25].

Recently, there has been significant interest in bursty event detection [26–29]. Bursty event detection models an event in text streams as a bursty activity, with certain features rising sharply in frequency (i.e., bursty features) as the event emerges [30–33]. Different from traditional retrospective detection and online detection methods, these methods group bursty features with identical trends to form events. There are two main issues related to bursty event detection, including bursty features identification and grouping bursty features into bursty events. Kleinberg [34] modeled the stream and extracted bursty features using infinite-state automation. He et al. [35] modeled aperiodic features with Gaussian density and periodic features with Gaussian mixture densities. An unsupervised greedy event detection algorithm was used to detect both aperiodic and periodic events. Fung et al. [31] also grouped bursty features to find bursty events. They identified a bursty feature by its distribution.

Different from the above work, we present a new method for online event detection in news streams. We represent news documents based on word embeddings which avoids high dimension and sparse semantics of TF-IDF vector representation. Then we use an adaptive online clustering algorithm which can automatically detect noisy events and further process them.

6. Conclusion and future work

In this paper, we propose a novel adaptive online event detection method for news streams. In our method, we present a novel document representation method based on word embeddings, which reduces the dimensionality of traditional document representation (TF-IDF vectors) and alleviates the sparse semantics, and thus improves the efficiency and accuracy. Based on our document representation, we propose a new adaptive online clustering method for online news event detection, which improves both the precision and recall by using time slicing and merging respectively. The resulted events are further improved by an adaptive post-processing step which can automatically detect noisy events and further process these events. Experiments on standard datasets and a real-world dataset show that our proposed adaptive online event detection method significantly improves the performance of event detection in terms of both efficiency and accuracy compared to state-of-the-art methods.

In future work, we will examine how to better incorporate the information of named entities into our approach, considering the importance of named entities such as persons, locations, organizations, etc. to further improve the performance of our adaptive online event detection method.

Acknowledgments

The work is supported by 973 Program (No. 2014CB340504), NSFC key project (No. 61533018, 61661146007), Fund of Online Education Research Center, Ministry of Education (No. 2016ZD102), and THUNUS NExT Co-Lab.

References

[1] J. Allan, R. Papka, V. Lavrenko, On-line new event detection and tracking, in: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 1998, pp. 37–45.
[2] J. Allan, S. Harding, D. Fisher, A. Bolivar, S. Guzman-Lara, P. Amstutz, Taking topic detection from evaluation to practice, in: System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on, IEEE, 2005. pp. 101a.
[3] R. Nallapati, A. Feng, F. Peng, J. Allan, Event threading within news topics, in: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management, ACM, 2004, pp. 446–453.
[4] Y. Yang, T. Pierce, J. Carbonell, A study of retrospective and on-line event detection, in: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 1998, pp. 28–36.
[5] Z. Li, B. Wang, M. Li, W.-Y. Ma, A probabilistic model for retrospective news event detection, in: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2005, pp. 106–113.
[6] J. Allan, J.G. Carbonell, G. Doddington, J. Yamron, Y. Yang, Topic detection and tracking pilot study final report (1998).
[7] W. Lam, H. Meng, K. Wong, J. Yen, Using contextual analysis for news event detection, Int. J. Intell. Syst. 16 (4) (2001) 525–546.
[8] Y. Yang, J. Zhang, J. Carbonell, C. Jin, Topic-conditioned novelty detection, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2002, pp. 688–693.
[9] K. Zhang, J. Zi, L.G. Wu, New event detection based on indexing-tree and named entity, in: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2007, pp. 215–222.
[10] M. Cordeiro, Twitter event detection: combining wavelet analysis and topic inference summarization, Doctoral symposium on Informatics Engineering, 2012.
[11] Y. Yang, J.G. Carbonell, R.D. Brown, T. Pierce, B.T. Archibald, X. Liu, Learning approaches for detecting and tracking news events, IEEE Intell. Syst. Appl. 14 (4) (1999) 32–43.
[12] X.-Y. Dai, Q.-C. Chen, X.-L. Wang, J. Xu, Online topic detection and tracking of financial news based on hierarchical clustering, in: Machine Learning and Cybernetics (ICMLC), 2010 International Conference on, 6, IEEE, 2010, pp. 3341–3346.
[13] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems, 2013, pp. 3111–3119.
[14] X. Dai, Y. Sun, Event identification within news topics, in: Intelligent Computing and Integrated Systems (ICISS), 2010 International Conference on, IEEE, 2010, pp. 498–502.
[15] L. Hou, J. Li, Z. Wang, J. Tang, P. Zhang, R. Yang, Q. Zheng, Newsminer: multifaceted news analysis for event search, Knowl. Based Syst. 76 (2015) 17–29.
[16] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent dirichlet allocation, J. Mach. Learn. Res. 3 (January 2003) 993–1022.
[17] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: Proceedings of the 31st International Conference on Machine Learning (ICML-14), 2014, pp. 1188–1196.
[18] A. Strehl, J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. 3 (December 2002) 583–617.
[19] J. Allan, H. Jin, M. Rajman, C. Wayne, D. Gildea, V. Lavrenko, R. Hoberman, D. Caputo, Topic-based novelty detection: 1999 summer workshop at clsp, final report, 1999.
[20] G. Kumaran, J. Allan, Text classification and named entities for new event detection, in: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2004, pp. 297–304.
[21] N. Keane, C. Yee, L. Zhou, Using topic modeling and similarity thresholds to detect events, in: Proceedings of the 3rd Workshop on EVENTS at the NAACL-HLT, 2015, pp. 34–42.
[22] R. Long, H. Wang, Y. Chen, O. Jin, Y. Yu, Towards effective event detection, tracking and summarization on microblog data, in: International Conference on Web-Age Information Management, Springer, 2011, pp. 652–663.
[23] J. Weng, B.-S. Lee, Event detection in twitter., ICWSM 11 (2011) 401–408.
[24] H. Becker, M. Naaman, L. Gravano, Learning similarity metrics for event identification in social media, in: Proceedings of the third ACM international conference on Web search and data mining, ACM, 2010, pp. 291–300.
[25] M. Cordeiro, Twitter event detection: combining wavelet analysis and topic inference summarization, Doctoral Symposium on Informatics Engineering, 2012.
[26] Y. Zhu, D. Shasha, Statstream: Statistical monitoring of thousands of data streams in real time, in: Proceedings of the 28th International Conference on Very Large Data Bases, VLDB Endowment, 2002, pp. 358–369.
[27] M. Vlachos, C. Meek, Z. Vagena, D. Gunopulos, Identifying similarities, periodicities and bursts for online search queries, in: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, ACM, 2004, pp. 131–142.
[28] A. Bulut, A.K. Singh, A unified framework for monitoring data streams in real time, in: Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on, IEEE, 2005, pp. 44–55.
[29] Z. Yuan, Y. Jia, S. Yang, Online burst detection over high speed short text streams, in: International Conference on Computational Science, Springer, 2007, pp. 717–725.
[30] J. Kleinberg, Bursty and hierarchical structure in streams, in: Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2002, pp. 91–101.
[31] G.P.C. Fung, J.X. Yu, P.S. Yu, H. Lu, Parameter free bursty events detection in text streams, in: Proceedings of the 31st International Conference on Very Large Data Bases, VLDB Endowment, 2005, pp. 181–192.
[32] Q. He, K. Chang, E.-P. Lim, Analyzing feature trajectories for event detection, in: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2007, pp. 207–214.
[33] W. Chen, C. Chen, L.-j. Zhang, C. Wang, J.-j. Bu, Online detection of bursty events and their evolution in news streams, J. Zhejiang Univ.–Sci. C 11 (5) (2010) 340–355.
[34] J.M. Kleinberg, Bursty and hierarchical structure in streams, in: Proceedings of the Eighth SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 91–101.
[35] Q. He, K. Chang, E.-P. Lim, J. Zhang, Bursty feature representation for clustering text streams, in: Proceedings of the 2007 SIAM International Conference on Data Mining, SIAM, 2007, pp. 491–496.