Expert Systems With Applications 115 (2019) 27–36

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Emerging topic detection in twitter stream based on high utility pattern mining

Hyeok-Jun Choi, Cheong Hee Park*

Department of Computer Science and Engineering, Chungnam National University, Daejeon, Republic of Korea

ARTICLE INFO

Article history:
Received 25 January 2018
Revised 24 July 2018
Accepted 25 July 2018
Available online 26 July 2018

Keywords:
Frequent pattern mining
High utility pattern mining
Topic detection
Twitter stream

ABSTRACT

Among internet and smart device applications, Twitter has become a leading social media platform, disseminating online events occurring in the world on a real-time basis. Many studies have been conducted to identify valuable information on Twitter. Recently, Frequent Pattern Mining has been applied for topic detection on Twitter. In Frequent Pattern Mining, a topic is considered to be a group of words that appear simultaneously, however, the method only considers the frequency of words, and their utility for topic detection is not considered in the process of pattern generation. In this paper, we propose a method to detect emerging topics on Twitter based on High Utility Pattern Mining (HUPM), which takes frequency and utility into account at the same time. For a chunk of tweets by time-based windowing on the Twitter stream, we define the utility of words based on the growth rate in frequency and find groups of words with high frequency and high utility by HUPM. For post-processing to extract actual topic patterns from candidate topic patterns generated by HUPM, an efficient data structure called Topic-tree (TP-Tree) is also proposed. Experimental results demonstrated the effectiveness of the proposed method, which showed superior performance and shorter running time than other tested topic detection methods. In particular, the proposed method showed a 5% higher topic recall than the other compared methods for the three datasets used.

1. Introduction

As the applications and uses of the internet continue to advance, social media applications including Twitter and Facebook have grown rapidly. Twitter has become one of the most widely employed social media applications, because of its unique features, including a simple interface and limit on the number of characters per posting. In 2016, Twitter was used by 300 million users per month (Statista, 2017), and more than 500 million new tweets were produced per day (Sayce, 2016). By easily accessing the Twitter service via smart devices, Twitter users spread online events occurring in many parts of the world on a real-time basis. Focusing on this feature of Twitter, various studies have been conducted in an effort to reveal valuable information from Twitter traffic, using methods such as Sentiment Analysis or Natural Disaster Prediction (Ibrahim, Abdillah, Wicaksono, & Adriani, 2015; Sakaki, Okazaki, & Matsuo, 2010). In particular, many studies are being actively pursued to find the answer to the question, "What is the hot issue right now?"

The process of extracting and summarizing trending issues in the form of useful information is called topic detection. Most previous topic detection methods have been designed for analyzing for a set of documents with a relatively formulated frame and a long sentence structure, such as news articles (He, Chang, & Lim, 2007; Wang, Zhang, Ru, & Ma, 2008). However, such conventional approaches are not applicable to the documents produced by Twitter, which consist of extremely short sentences containing misspelled words. Moreover, since tweets often include not only sentences but also special characters or URLs, the ratio of tweets including useful information is very low in comparison with the volume of all tweets. In addition, it is difficult to identify correlations between tweets due to the diversity of issues, and this makes it hard to distinguish topics. Ultimately, a real-time topic detection method is needed because large volume of tweets are produced in a very short time.

Among the various methods attempted for topic detection in Twitter, in the feature-pivot based topic detection method, a topic is considered to be a group of words that occur simultaneously. The feature-pivot methods have various approaches for identifying those groups of essential words. Recently, Frequent Pattern Mining (FPM) has been applied for topic detection in Twitter. The goal of the FPM in topic detection is to find patterns of words which occur

* Corresponding author.
E-mail address: cheonghee@cnu.ac.kr (C.H. Park).

https://doi.org/10.1016/j.eswa.2018.07.051
0957-4174/

frequently in a document set. The method called SFPM (Soft FPM) applies a modified version of the FPM by mitigating the requirement that all items must be frequent in the pattern (Petkos, Papadopoulos, Aiello, Skraba, & Kompatsiaris, 2014). Gaglio, Lo *Re*, and Morana (2015) extended SFPM using Named Entity Recognition. In Huang, Peng, and Wang 2015, topic detection is conducted by clustering all the patterns generated by the FP-Growth algorithm.

FPM generates a pattern of all the items with high frequency in the entire transaction (Han, Pei, & Yin, 2000, Cormode & Hadjieleftheriou, 2008). In FPM, all the items are considered to have equal value or utility. However, there are many situations where the utility of each item is not equal. For example, in market basket transactions, the utility of an item can be defined as the quantity of an item which a customer purchases, or the price of an item. Recently, High Utility Pattern Mining (HUPM) has been introduced. Its goal is to find itemsets that have high frequency and high utility at the same time (Liu & Qu, 2012).

In this paper, we propose a topic detection method for Twitter by using High Utility Pattern Mining. HUPM is generally suitable for a batch of data. In order to apply HUPM to the Twitter streams, we adapt a sliding window technique and construct a transaction table from the tweets within a window where one tweet is considered as a transaction and words in tweets are taken as items. In order to find an emerging topic in the Twitter stream, we need to detect words that appear in many tweets. Another important characteristic of words constituting an emerging topic is the rapid increase in their appearance. This phenomenon is detectable on Twitter due to its unique characteristics, such as the limited number of words in a tweet. We define the utility of a word based on the growth in appearance frequency and apply HUPM in order to find a set of words that have high frequency and high utility. We also present a method to dynamically determine the minimum utility threshold. Lastly, post-processing was performed to extract the actual topic patterns from the candidate topic patterns generated through HUPM.

The contribution of this paper can be summarized as follows:

- We propose a topic detection method for Twitter using High Utility Pattern Mining. The proposed method considers both the frequency of words over tweets and the utility of words, which is defined based on the growth rate in appearance frequency.
- A technique to dynamically determine the minimum utility threshold for each chunk of tweets is presented.
- We define a Topic-tree (TP-Tree) for post-processing to extract actual topic patterns from candidate topic patterns generated by HUPM.
- The experimental results demonstrate that the proposed topic detection method has superior topic recall performance and shorter running time than the compared methods.

The paper is organized as follows. In Section 2, related work is reviewed. Section 3 describes the basic concepts of HUPM. In Section 4, we present a topic detection method for Twitter streams based on HUPM. Section 5 describes the experimental procedures and results. In Section 6, the conclusions are given.

2. Related work

Topic detection methods are generally classified into three categories. First, the feature-pivot based methods finds word groups appearing simultaneously in the document set (Aiello et al., 2013; Gaglio et al., 2015; Huang et al., 2015; Li, Sun, & Datta, 2012; Mathioudakis & Koudas, 2010; Petkos et al., 2014; Weng & Lee, 2011; Zhang, Yoshida, Tang, & Wang, 2010). Secondly, the document-pivot based methods compare the similarities between documents, and then group similar documents (Becker, Naaman, & Gravano, 2011; O' Connor, Krieger, & Ahn, 2010; Petrovic, Osborne, & Lavrenko, 2010; Phuvipadawat & Murata, 2010; Sankaranarayanan, Samet, Teitler, Lieberman, and Sperling (2009); Zhou & Chen, 2014). Lastly, the probabilistic topic model detects topics by predicting the probability distribution of words with respect to topics and that of topics with respect to documents (Blei, Ng, & Jordan, 2003; Hofmann, 1999; Kim, Park, Lu, & Zhai, 2012; Quercia et al., 2012).

2.1. Feature-pivot methods

In feature-pivot based methods, a topic is expressed as a group of words, and the goal is to determine the word groups that appear simultaneously in a document set. The essential words are extracted based on the frequency and burstiness of the words. Then, the similarity between the essential words is measured to group the words. The individual groups represent specific topics. One of the methods for selecting the essential words is to focus on the bursty words whose frequency has greatly increased. In the Twitter topic detection method of (Mathioudakis & Koudas, 2010), bursty words having a frequency higher than usual were detected, and more emerging words were identified in reference to the detected bursty words. In Weng and Lee (2011), wavelet analysis was employed for the selection of bursty words, and community detection was used for the grouping of the selected bursty words. Another method for selecting the essential words is to focus on the simultaneous appearance of a part of a sentence or word group through a frequent pattern or n-gram. In Zhang et al. (2010), a clustering method was used to find the frequent patterns from a document set. In Gaglio et al. (2015), Huang et al. (2015), Petkos et al. (2014), FPM was applied for topic detection. In Aiello et al. (2013), Li et al. (2012), the n-grams of words were generated in Tweets and clustering of the generated n-grams was performed. The individual clusters represent specific topics. In Aiello et al. (2013), the df–idf_t score was defined for each n-gram of the current time slot i based on its document frequency for this time slot and penalized by the logarithm of the average of its document frequencies in previous t time slots, such as

$$df - idf_t = \frac{df_i + 1}{\log\left(\frac{\sum_{j=i}^{t} df_{i-j}}{t} + 1\right) + 1}.$$

The clusters with the highest df–idf_t scores represent topics. While the df–idf_t score reflects the changing rate of document frequencies, our proposed method uses the increasing rate of term frequencies between adjacent time slots in order to define the utility of words. Instead of using n-grams of words, our method takes into account the simultaneous co-occurrence of terms by using high utility pattern mining. We also present a TP-Tree for efficient post-processing to remove redundancy in the found topics. In Erra, Senatore, Minnella, and Caggianese (2015), an approximate version of the TF–IDF measure and a parallel implementation of the approximate TF–IDF calculation using GPUs were proposed. By using the hashtags as a document identifier, tweets were transformed to the pairs of ⟨term, document⟩. For a collection of ⟨term, document⟩ pairs, top-k frequent pairs by the approximate TF–IDF measure were extracted and trending topics were analyzed by the most frequent hashtag-term pairs. While hashtags were used as a document identifier for counting term and document frequencies in Erra et al. (2015), our method treats hashtag words as usual terms and popular hashtag words can be used as the words representing emerging topics.

2.2. Document-pivot methods

In document-pivot based methods, a topic consists of a set of documents. When documents are flowing in, the similar-

ity between individual documents or document groups is compared, and the documents having a similarity value higher than a specific threshold are grouped together. Each of the generated groups represents a topic. The document-pivot based methods are generally characterized according to the method used to represent documents and measure the similarity between documents. In O' Connor et al. (2010), Phuvipadawat and Murata (2010), Sankaranarayanan, Samet, Teitler, Lieberman, and Sperling (2009), the tweets are represented as vectors using Term Frequency-Inverse Document Frequency (TF-IDF). In Becker et al. (2011), the features for tweets such as the number of followers or mentions were used, as well as TF-IDF. Petrovic et al. (2010) proposed a method of finding the clusters of similar documents by modifying the Locality Sensitive Hashing (LSH). In Zhou and Chen (2014), a graphical model was proposed for measuring the similarity between documents on the basis of the text, time, and geographic information included in the documents. However, since tweets are short sentences, the vectors produced by using TF-IDF may be sparse vectors. Therefore, the performance of document-pivot methods is difficult to guarantee.

2.3. Probabilistic topic model

In a probabilistic topic model, topics are detected by approaching the topic detection process as a probabilistic inference problem. A topic is expressed as a probability distribution of words and documents are considered to be the probability distributions of topics. Probabilistic topic models have recently been used extensively to detect topics from document sets. Latent Semantic Analysis (LDA) and Probabilistic Latent Semantic Analysis (PLSA) are the representative probabilistic topic models (Blei et al., 2003; Hofmann, 1999; Quercia, Askham, & Crowcroft, 2012). In Kim et al. (2012), a topic modeling method was suggested where FPM and a probabilistic topic model were combined. It showed that topic detection performance was better than conventional LDA and PLSA. However, topic detection methods using a probabilistic topic model have difficulty in setting parameter values and may not be suitable for real-time topic detection due to their long runtime.

3. High utility pattern mining

While FPM finds a set of frequently appearing items from entire transactions, the goal of HUPM is to find a set of items that are frequently appearing and highly valuable at the same time. We applied HUPM for emerging topic detection in Twitter. In this section, we briefly describe the basic concepts and definitions of HUPM based on the paper by Liu and Qu (2012), which proposed an efficient algorithm for mining high utility itemsets without candidate generation.

Definition 1. *[Transaction table].* Given a set of items, a transaction table contains subsets of items. Unlike transactions used in FPM, each item in a transaction is accompanied by the item frequency. If a tweet is considered to be a transaction, words in a tweet can be treated as items together with the word frequency in the tweet. Table 1 gives an example of a transaction table which contains seven transactions with items *a, b, c, d, e, f, g*.

Definition 2. *[External utility, Internal utility, Utility].* An external utility for item i means the value of the item, expressed as $eu(i)$. Table 2 gives an example of a utility table showing the external utility of individual items. An internal utility of item i represents the frequency of the item in the transaction, expressed as $iu(i, T)$. With $eu(i)$ and $iu(i, T)$, the utility of item i for the transaction T, $u(i, T)$, is defined as follows:

$$u(i, T) = iu(i, T) \times eu(i) \tag{1}$$

Table 1
An example of a transaction table which is composed of items *a,b,c,d,e,f,g.*. The last column shows the transaction utility (TU) of each transaction.

Tid	Transaction	TU
T_1	{*a*:2, *b*:1, *c*:3, *d*:1}	25
T_2	{*c*:2, *e*:1, *f*:1}	11
T_3	{*a*:5, *b*:1, *d*:1, *e*:2}	32
T_4	{*a*:3, *b*:1, *c*:1, *f*:2}	23
T_5	{*d*:1, *e*:1, *g*:5}	13
T_6	{*b*:1, *c*:1, *d*:1, *g*:2}	13
T_7	{*a*:4, *c*:2, *d*:1}	28

Table 2
An example of a utility table showing the external utility (eu) of individual items.

Item	*a*	*b*	*c*	*d*	*e*	*f*	*g*
External utility	4	2	3	6	2	3	1

Table 3
Transaction-weighted utility of 1-itemsets.

Itemset	{*a*}	{*b*}	{*c*}	{*d*}	{*e*}	{*f*}	{*g*}
TWU	108	93	100	111	56	34	26

Definition 3. *[Itemset utility, Transaction utility, Transaction-weighted utility].* If X denotes a subset of the items included in transaction T, the utility of the itemset X, $u(X, T)$, the transaction utility of the transaction T, $tu(T)$, and the transaction-weighted utility, $twu(X)$, are defined as follows:

$$u(X, T) = \sum_{i \in X} u(i, T), \tag{2}$$

$$tu(T) = \sum_{i \in T} u(i, T), \tag{3}$$

$$twu(X) = \sum_{X \subset T} tu(T). \tag{4}$$

HUPM generates a high utility pattern which is an itemset X satisfying $twu(X) \geq$ min-*util*. The value min-*util* needs to be determined by the user in advance. TWU (transaction-weighted utility) is an important property satisfying the downward closure, that is, the TWU of a set is less than or equal to the TWU of a subset of it. TWU is used for pattern generation and pruning in HUPM (Liu, Liao, & Choudhary, 2005).

The last column in Table 1 shows the transaction utility (TU) of each transaction calculated by Eq. (3). For example, the TU of $T_2 = \{c, e, f\}$ is obtained as $tu(T_2) = 2 \times 3 + 1 \times 2 + 1 \times 3 = 11$. Table 3 shows the TWU of the 1-itemsets. The TWU of the 1-itemset $\{a\}$ is $twu(\{a\}) = tu(T_1) + tu(T_3) + tu(T_7) = 108$. If the min-*util* is set as 50, all itemsets including $\{f\}$ or $\{g\}$ are pruned. As a result, the generated high utility pattern is (c, a), (c, a, d), (a), (a, d).

4. Topic detection based on high utility pattern mining

We applied HUPM for topic detection in Twitter streams. Fig. 1 shows a flowchart of the proposed method. The entire process includes the computation of utility for words appearing in tweets, the determination of min-*util*, the generation of candidate topic patterns by HUPM, and post-processing for the extraction of topic patterns.

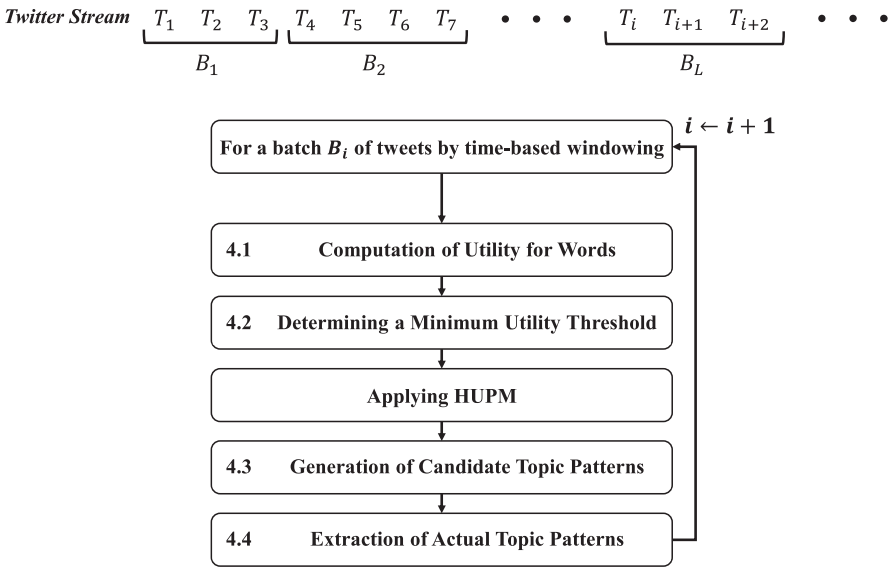


Fig. 1. The flowchart of the proposed method for emerging topic detection in a Twitter stream.

4.1. Computation of utility for words

Assume that tweets generated in a time order are denoted as T_i and the Twitter streams as $TS = T_1, T_2, T_3, \ldots$. If a non-overlapping sliding window technique is applied in a certain time interval, TS is represented as a sequence of batches, B_1, B_2, B_3,... Let the current position be at batch B_L, as shown in Fig. 1. The individual tweet included in B_L is reformed as a form of a transaction given in Table 1, where the words in the tweets are accompanied by the word frequency in the tweet.

The external utility represents the value of the individual words in detecting new topics. The high utility patterns produced by HUPM are significantly dependent on the external utility values. Since the words generated in each of the batches and their frequencies change between adjacent batches, the utility of words should be computed in each batch.

Since the maximum number of characters that may be included in a tweet is limited, the frequency of specific words related to an issue increases rapidly. Let us denote the frequency of word i for batch B_t as $f(B_t,i)$. We can define the difference in the frequency of word i between the current batch B_L and the previous batch B_{L-1} as in Eq. (5).

$$diff(i) = f(B_L, i) - f(B_{L-1}, i) \tag{5}$$

If $diff(i) > 0$, the frequency of word i is currently increasing. If $diff(i) < 0$, the frequency of word i is currently decreasing. On the other hand, the rate of frequency increase of word i between batch B_L and the previous batch B_{L-1}, $rate(i)$, can be calculated as

$$rate(i) = \frac{f(B_L, i) + 1}{f(B_{L-1}, i) + 1}. \tag{6}$$

To prevent the denominator in Eq. (6) becoming zero, the value of 1 is added to the numerator and to the denominator. If $rate(i) > 1$, the frequency of word i is currently increasing. If $rate(i) < 1$, the frequency of word i is currently decreasing.

Using diff(i) and rate(i), the external utility for word i included in batch B_L is defined as in Eq. (7).

$$eu(i) = \begin{cases} \text{diff}(i) \times \log(\text{rate}(i)) & if\ \text{diff}(i) > 0, \\ 0 & else \end{cases} \tag{7}$$

Since $diff(i) \leq 0$ implies that the frequency of word i is either unchanging or decreasing, the utility for the word i is set to be zero, under the assumption that such a word is unlikely to be an essential word for an emerging topic. The value of $rate(i)$, calculated as a ratio of word frequency, may be extremely large. Therefore, a logarithm of $rate(i)$ is used in the calculation of $eu(i)$.

4.2. Determining a minimum utility threshold

In HUPM, a set X of words is generated as a high utility pattern if $twu(X) \geq$ min-*util*. If the min-*util* is too small, too many patterns may be generated. On the contrary, if the min-*util* is too large, almost no patterns may be generated. In addition, the min-*util* plays a key role in determining the length of the generated patterns. For the patterns generated by HUPM as topic patterns, an appropriate number of words should be included in the patterns. A fixed min-*util* may not be used for all the batches because the transaction information included in the batches is varied. Methods for determining the min-*util* have been studied under the theme of top-k high utility pattern mining (Ryang & Yun, 2015; Tseng, Wu, Fournier-Viger, & Yu, 2016; Wu, Shie, Tseng, & Yu, 2012), but these methods cannot be applied to data streams on a real-time basis because the methods take a long time in determining the min-*util*.

Now we describe how to set the min-*util* in batch B_L dynamically. Denoting the number of all tweet posts containing the words in X as $s(X)$, the length $l(T)$ of a tweet post T is defined as the sum of the internal utilities of the words included in T, $\sum_{i \in T} iu(i, T)$. With $s(X)$ and $l(T)$, the calculation of TWU for X, defined in Eq. (4), can be modified as shown in the following equation

$$twu(X) = \frac{\sum tu(T)}{\sum l(T)} \times \frac{\sum l(T)}{s(X)} \times s(X). \tag{8}$$

The summation in Eq. (8) is performed over the tweets T including the words in X and we omit the subscript of summation when it is clear. If the three terms on the right side of Eq. (8) are respectively denoted as α, β, and γ, then the first term $\alpha = \frac{\sum tu(T)}{\sum l(T)}$ means the utility average of words which belong to a tweet containing X. The second term $\beta = \frac{\sum l(T)}{s(X)}$ is the average length of tweets which contain X, and the third term $\gamma = s(X)$ is the number of tweets containing X.

The min-*util* is calculated in the following process. We select the words with the highest external utility values among all the words included in the current batch B_L, since the words having high external utility are highly probable to be the essential words

constituting a topic. We set the parameter ρ as the ratio of selected words. However, the total number of words may be greatly different between batches, and the number of selected words may be too small or too great. Therefore, a lower-bound ω and an upper-bound θ for the number of selected words are established for word selection. If we denote the number of all the words in batch B_L as S, the number of selected words is determined to be

$$\textit{the number of selected words} = \begin{cases} \omega & \text{if } \rho S < \omega, \\ \rho S & \text{if } \omega \le \rho S \le \theta, \\ \theta & \text{if } \rho S > \theta. \end{cases} \quad (9)$$

Now we calculate α, β, and γ for selected words. If the averages of α, β, and γ of the selected words are respectively denoted as $avg(\alpha)$, $avg(\beta)$, and $avg(\gamma)$, the min-*util* is calculated as in the following Equation.

$$\textit{min-util} = avg(\alpha) \times avg(\beta) \times avg(\gamma) \quad (10)$$

In Eq. (10), the min-*util* reflects the TWU of X including the essential words. In addition, since the calculation is carried out each time when the batch is changed, the min-*util* is dynamically determined.

4.3. Generation of candidate topic patterns

For the batch, B_L, the information obtained in Sections 4.1 and 4.2 is used as the input for the HUPM to generate a high utility pattern X satisfying the condition $twu(X) \geq$ min-*util*. In the HUPM described in Liu and Qu (2012), the TWU of the set of a single word is initially calculated, and then the words are sorted in an ascending order of TWU. Subsequently, the words having a TWU smaller than the min-*util* are removed from the tweet, and the entire tweets are reconstructed with the remaining words sorted in an ascending order. After that, high utility patterns are generated by using a data structure called utility-lists. However, most of the patterns generated in this manner include redundant patterns, where some patterns of short length appear repeatedly in the patterns of long length. We call the patterns generated by HUPM candidate topic patterns and apply post-processing to eliminate the redundant patterns.

4.4. Extraction of actual topic patterns

To effectively remove the redundancy from the candidate topic patterns, we construct a tree, called a TP-Tree (Topic-tree), that compactly represents the candidate topic patterns. The constructed TP-Tree is an adaptation of the ideas for FP-Tree (Han et al., 2000). As in FP-Tree, the header table indicating the tree node is maintained to easily investigate the redundancy between a newly inserted candidate pattern and the existing patterns.

Each node in the TP-Tree contains the label of an item and a reference field to a node. Initially, the TP-Tree contains only the root node represented by the "*blank*" symbol and the header table is empty. A candidate topic pattern (i_1,i_2,..., i_n), which is an ordered list of words, is inserted into the TP-Tree as follows:

(1) If the node for i_1 does not exist in the header table, a path from the root node, blank $\rightarrow i_1 \rightarrow i_2 \rightarrow \ldots \rightarrow i_n$, is created. Go to the step (3) for the update of the header table.
(2) If the node for i_1 exists in the header table, the following actions are taken while traversing along the linked list L starting from the node i_1 of the header table.
  (2-1) If i_2,..., i_n are found in order along the path beginning from a node on L, the insertion is finished. However, i_2,..., i_n do not need to be found consecutively along the path. (Fig. 2(b) shows an example for the case where i_1,..., i_n exist discontinuously along the path.).
  (2-2) If i_2,..., i_n are not found on the paths beginning from any node on L, a path from the root node, blank $\rightarrow i_1 \rightarrow i_2 \rightarrow \ldots \rightarrow i_n$, is created, sharing the nodes of the common prefix terms with the existing root-leaf paths. Go to step (3) for the update of the header table.
(3) For a newly created node for item i_k from $k = 1$ to n,
  (3-1) If the node for i_k does not exist in the header table, a new node for i_k is added to the header table and a link from the node i_k of header table to the node i_k in the tree is established.
  (3-2) If the node for i_k already exists in the header table, a link is established from the last node on the linked list L to node i_k in the tree.

If the insertion order of candidate topic patterns is changed, the TP-Tree can be slightly changed. However, HUPM generates candidate topic patterns by a certain rule and we can always get the same TP-Tree by inserting all the candidate topic patterns in the order generated by HUPM. The time complexity for the construction of the TP-Tree can be described as $O(nlw)$, where n is the number of candidate topic patterns, l is the maximum length of candidate topic patterns, and w is the width of the TP-Tree. For each candidate topic pattern, while traversing along linked list L starting from the header table, the search for the pattern is repeated along root-leaf paths.

Fig. 2 illustrates the construction process of the TP-Tree. Suppose that the candidate topic patterns, (*e, c, b, d*), (*e, c, d*), (*c, d, f*), and (*c, b, d, f*) are inserted in the order.

- Fig. 2(a) shows how the first pattern (*e, c, b, d*) is added to the tree. Since there is no node *e* in the initial header table, a path from the root, $blank \rightarrow e \rightarrow c \rightarrow b \rightarrow d$, is added and the header table is updated.
- Fig. 2(b) shows how the second pattern (*e, c, d*) is added to the tree. Since node *e* exists in the header table, the presence of (*c, d*) in any path along the linked list of node *e* is checked and (*e, c, d*) is not added.
- Fig. 2(c) shows how the third pattern (*c, d, f*) is added to the tree. Since node *c* exists in the header table, the presence of (*d, f*) in any path along the linked list of node *c* is checked. In the path, $blank \rightarrow e \rightarrow c \rightarrow b \rightarrow d$, node *d* exists but not *f*. Hence, a path from the root $blank \rightarrow c \rightarrow d \rightarrow f$ is added and the header table is updated.
- Fig. 2(d) shows how the fourth pattern (*c, b, d, f*) is added to the tree. While (*b, d*) exists in the path, $blank \rightarrow e \rightarrow c \rightarrow b \rightarrow d$, there is no *f*. While (*d, f*) exists in the path, $blank \rightarrow c \rightarrow d \rightarrow f$, there is not *b*. Hence, the path is from the root blank $\rightarrow c \rightarrow b \rightarrow d \rightarrow f$, sharing a common prefix node *c*.

TP-Tree looks similar to a Trie (Fredkin, 1960), also called digital tree, but the insertion process using the header table makes TP-Tree different from the Trie. For example, if the patterns (*a, b, c*) and (*a, c*) are inserted into the Trie, two paths of $a \rightarrow b \rightarrow c$ and $a \rightarrow c$ are generated. However, in the TP-Tree, only the path $a \rightarrow b \rightarrow c$ is generated because it uses the header table to check whether there are terms in the existing paths.

In the constructed TP-Tree, a path from the root node to the leaf node becomes a topic pattern. To prioritize the topic patterns, the utility of the pattern *p*, $PU(p)$ is defined as follows.

$$PU(p) = \sum_{i \in p} eu(i)$$

The *PU* means the sum of external utilities for the words included in the pattern. All the patterns are sorted in a descending order of *PU* values, and the top-*k* patterns are extracted as the final topic patterns. For each batch, a TP-Tree is constructed from

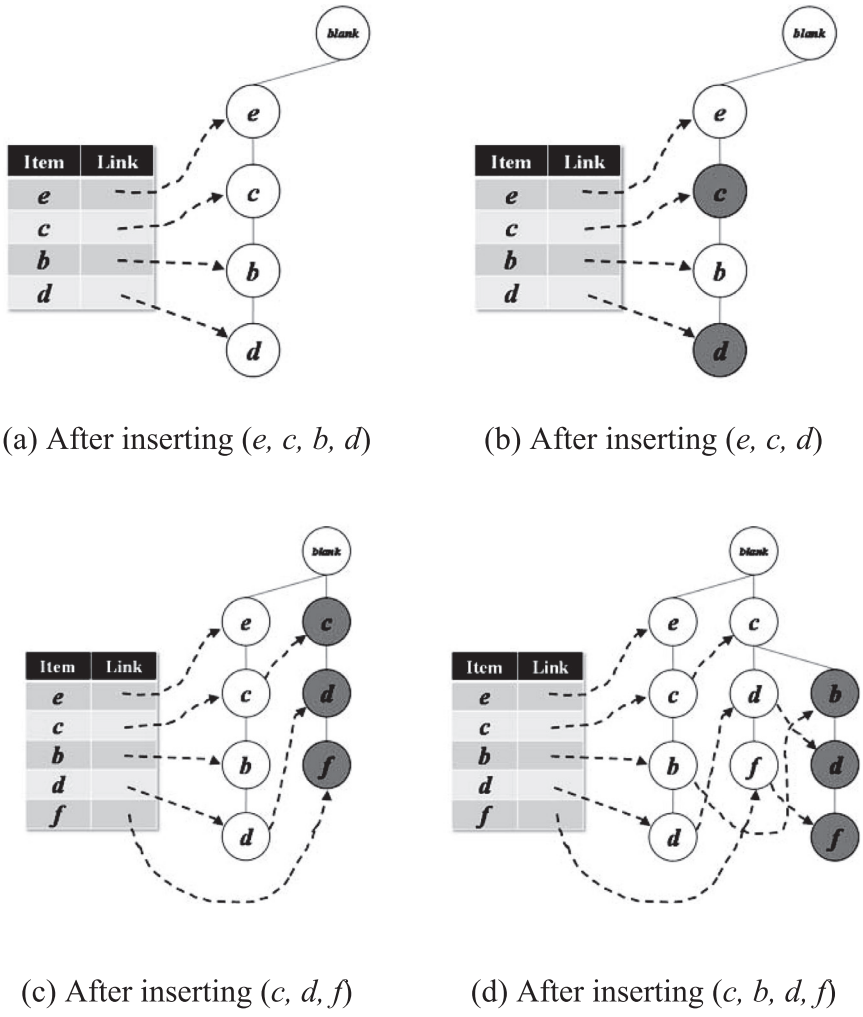


Fig. 2. Construction Process of TP-Tree from Candidate Topic Patterns where the candidate topic patterns, (*e, c, b, d*), (*e, c, d*), (*c, d, f*), and (*c, b, d, f*) are inserted in order.

candidate topic patterns generated by the process of Section 4.3. When the window moves along the data stream, a new TP-Tree is constructed.

5. Experimental results

5.1. Twitter data

Experiments were performed with the Twitter data used in Aiello et al. (2013) for topic detection.[1] Three types of datasets include tweets about the FA Cup Final (FA), Super Tuesday (ST), and US Elections (US) respectively, which were the three major events in 2012. Tweets were extracted from the public Twitter Streaming API[2] by using a set of filter keywords and hashtags chosen by experts. The tweets related to each event were partitioned into time slots, specifically one minute for the FA, 60 min for the ST, and 10 min for the US. The ground truth topics were extracted by using mainstream media reports. The published media report accounts of the events were reviewed and a set of stories was chosen to build a topic ground truth. The details for data construction can be found in Aiello et al. (2013). Three datasets are described below.

FA Cup Final (FA): This includes a total of 360 intervals, and each interval includes the tweets produced in one minute in the time order. Ground-truth data is also given which contains the information for 13 topics in 13 intervals.

Super Tuesday (ST): This includes a total of 24 intervals, and each interval has the tweets produced in 60 min in the time order. Ground-truth data contains 22 topics in 8 intervals.

US Elections (US): This data is about the election of the US president. The data includes a total of 216 intervals, and each interval includes the tweets produced in 10 min in the time order. The ground-truth information for the 64 topics in 26 intervals is given.

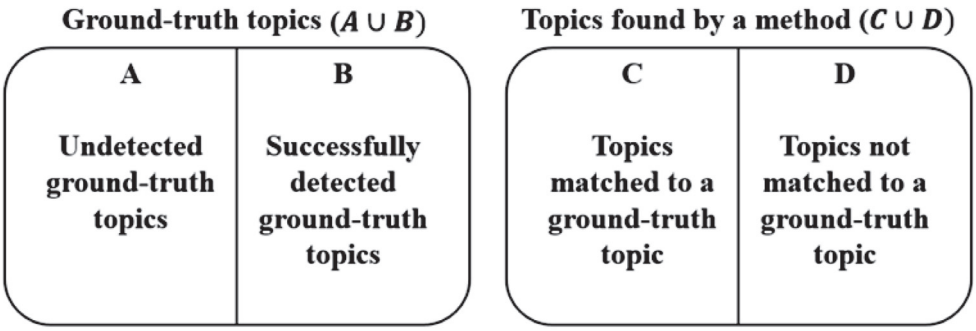
Ground-truth: This contains information about the topics that were actually raised. The ground-truth topic was expressed by a set of mandatory keywords and optional words. For a detected topic to be evaluated as a ground-truth topic, the topic should include all the essential keywords. An optional word is a word that is selectively needed to constitute a topic and does not affect the constitution of the ground-truth topics.

5.2. Data preprocessing

A tweet contains various elements such as hashtags, user mentions, and retweets, in addition to the sentences a user writes. Therefore, the elements necessary for topic detection should be extracted through pre-processing. All the tweets used in the experiment underwent preprocessing as follows:

- All the characters were changed to small letters.

[1] https://www.socialsensor.eu/results/datasets/72-twitter-tdt-datasets/.
[2] https://developer.twitter.com/en/docs/.



$$Topic\ recall = \frac{|B|}{|A \cup B|} \qquad Topic\ relevance = \frac{|C|}{|C \cup D|}$$

Fig. 3. The visual description for topic recall and topic relevance.

- The tweets that are collected through the Twitter Streaming API often include HTML tags such as , &, <, >, ", and so on. When a user writes a URL, it is converted to a form like "https://t.co/…". We changed the HTML tags to white-space and removed URLs included in the tweet.
- The hashtag reflects the characteristics of a specific issue, and the user mentions and retweets reflect information about a specific person. Thus, we performed tokenization including all of them. The tokenization process was conducted using Lucene's Standard Analyzer,[3] where stop-words and special characters including a hashtag indicator were removed and then the sentences were tokenized by white-spaces. Hence, the words in the hashtags composed of multiple words are also tokenized after the special character # is removed.

5.3. Measures for performance evaluation

The reference material (Aiello et al., 2013) provides not only the datasets but also scripts for evaluating of the topic detection performance. The topic detection performance was evaluated using the evaluation scripts, and topic recall, keyword precision, and keyword recall were used as the performance evaluation scales. F-measure combining keyword precision and keyword recall was given. We used micro-averaged precision, recall, and F-measure over all time intervals. Additionally, we measured the ratio of the topics evaluated as ground-truth topics among topics found by a topic detection algorithm.

Topic recall and topic relevance: When the topic found by a topic detection method included all the mandatory keywords specified for the ground-truth topic, it was considered that the ground-truth topic was successfully detected or the found topic was matched to the ground-truth topic. Topic recall is the ratio of the topics successfully detected among the ground-truth topics. Topic relevance is the ratio of the topics matched to some ground-truth topic among the topics found by a method. Fig. 3 describes topic recall and topic relevance. Note that the mapping between B and C might not be a one-to-one correspondence. For example, suppose that four ground-truth topics are given. If six topics among ten found topics are matched to three ground-truth topics, then $|A \cup B| = 4$, $|B| = 3$, $|C \cup D| = 10$, and $|C| = 6$. Hence, topic recall is 3/4 and topic relevance is 6/10.

Keyword precision and keyword recall: Keyword precision is the ratio of correctly detected keywords out of the total number of keywords for the found topics matched to some ground-truth topic. Correctly detected keywords mean that the keywords are contained in the matched ground-truth topic. Keyword recall refers to the ratio of correctly detected keywords over the total number of keywords in the ground-truth topics matched to some found topic. Keyword precision and recall are described in Eq. (11) by using the sets from Fig. 3.

$$\begin{aligned} keyword\ precision &= \frac{the\ detected\ keywords\ among\ the\ keywords\ for\ the\ topics\ in\ C}{the\ total\ number\ of\ keywords\ for\ the\ topics\ in\ C} \\ keyword\ recall &= \frac{the\ detected\ keywords\ among\ the\ keywords\ for\ the\ topics\ in\ B}{the\ total\ number\ of\ keywords\ for\ the\ topics\ in\ B} \end{aligned} \tag{11}$$

F-measure: From keyword precision and keyword recall, F-measure is computed as

$$F - measure = \frac{2 \times Keyword\ precision \times Keyword\ recall}{(keyword\ precision + Keyword\ recall)}$$

Topic recall can be used as a major standard to evaluate topic detection performance, and F-measure is compared when needed. The methods used to compare topic detection performance include feature-pivot methods, such as BNgram (Aiello et al., 2013), SFPM (Petkos et al., 2014), and Graph-based (Aiello et al., 2013), a document-pivot method Doc-Pivot (Petrovic et al., 2010), and a probabilistic topic model-based LDA (Teh, Newman, & Welling, 2007). The parameters used for the above methods were set as the default values of the evaluation script. Notably, the performance of the probabilistic topic-model based LDA can vary depending on the number of iterations. In the evaluation script, the number of iterations is set to 300.

5.4. Performance comparison

Table 4 compares the performance of the proposed method and the compared methods. In the proposed method, a ρ value, a lower-bound ω, and an upper-bound θ should be set up. In the present experiment, we tested with a setting of 27 combinations using the ρ values of 0.03, 0.05, 0.07, the ω values of 30, 50, 70, and the θ values of 300, 500, 700. In Table 4, we report the results obtained when the parameters were set as $\rho = 0.03$, $\omega = 50$, and $\theta = 300$. As shown in Table 4, the experimental results for the FA dataset indicate that the topic recall was highest for the proposed method (0.923). The topic recall of BNgram was 0.846, which was highest among the compared methods but lower than that of the proposed method by about 8%. The keyword precision and keyword recall were highest in SFPM, but the f-measure of SFPM was 0.460, which was just 4% higher than that of the proposed method

[3] https://lucene.apache.org/core/.

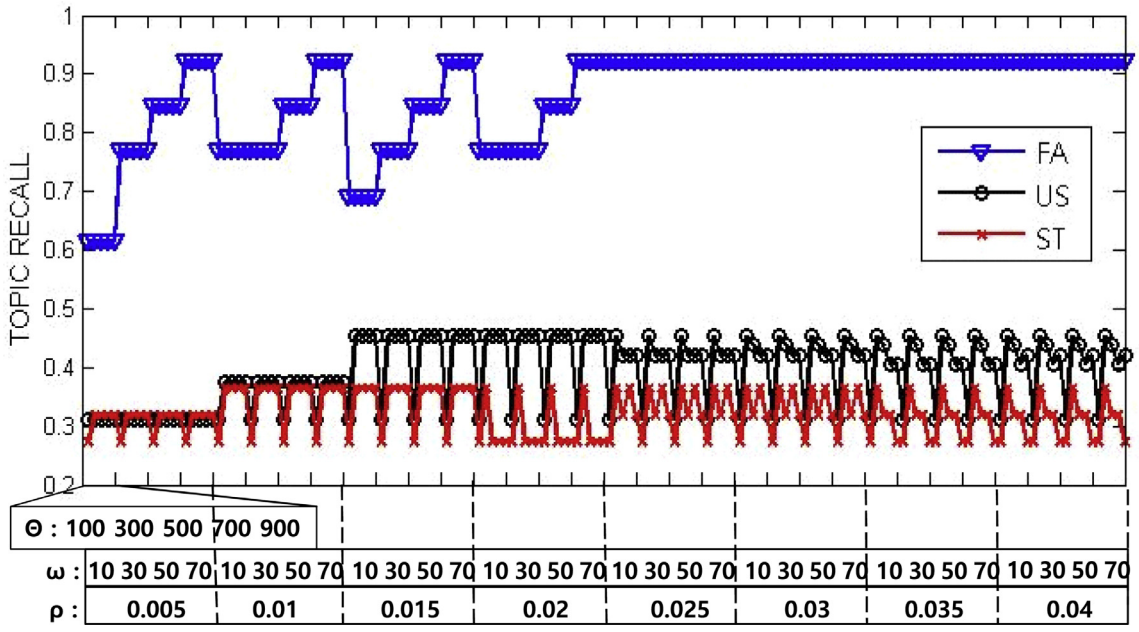


Fig. 4. The comparison of topic recall under the ρ values of 0.005–0.04, the ω values of 10, 30. 50. 70, and the θ values of 100, 300. 500, 700, 900.

Table 4
Performance comparison of the compared topic detection methods on three datasets.

FA Cup					
Method	Topic related		Keyword related		
	Recall	Relevance	Precision	Recall	F-measure
Proposed	**0.923**	**0.692**	0.320	0.600	0.418
BNgram	0.846	0.423	0.341	0.569	0.426
SFPM	0.538	0.538	**0.370**	**0.606**	**0.460**
G-Based	0.538	0.269	0.211	0.516	0.299
D-Pivot	0.538	0.269	0.283	0.515	0.366
LDA	0.385	0.192	0.260	0.565	0.356
Super tuesday					
Method	Topic related		Keyword related		
	Recall	Relevance	Precision	Recall	F-measure
Proposed	**0.364**	**0.313**	0.486	**0.708**	0.576
BNgram	0.318	0.100	**0.650**	0.605	**0.627**
SFPM	0.182	0.288	0.600	0.577	0.588
G-Based	0.000	0.000	0.000	0.000	0.000
D-Pivot	0.136	0.038	0.300	0.450	0.360
LDA	0.273	0.075	0.350	0.600	0.442
US elections					
Method	Topic related		Keyword related		
	Recall	Relevance	Precision	Recall	F-measure
Proposed	**0.453**	0.242	0.352	0.565	0.434
BNgram	0.328	0.085	**0.413**	0.529	**0.464**
SFPM	0.313	**0.385**	0.380	0.522	0.440
G-Based	0.031	0.008	0.182	**0.667**	0.286
D-Pivot	0.047	0.012	0.259	0.467	0.333
LDA	0.391	0.104	0.324	0.579	0.415

(0.418). In particular, the topic recall of the SFPM was about 40% lower than that of the proposed method.

Similarly, for the ST dataset, the topic recall of the proposed method was highest, at 0.364. The topic recall of BNgram was 0.318, which was highest among the compared methods but lower than that of the proposed method by about 5%. The keyword precision was highest in BNgram, but the keyword recall was highest in the proposed method at 0.708. The experimental results from the US Elections dataset show that the topic recall of the proposed method was highest at 0.453. Among the compared methods, the topic recall was highest in the order of LDA and BNgram. The topic recall of LDA was 0.391 which was lower than that of the proposed method by about 6%, and that of BNgram was 0.328 which was lower than that of the proposed method by about 13%. The keyword precision and keyword recall were highest in BNgram and Graph-based, respectively, at 0.413 and 0.667, but the F-measure of the methods was not significantly different from that of the proposed method.

5.5. Parameter sensitivity

In the proposed method, the ρ value, the lower-bound, ω, and the upper-bound, θ, need to be set up. Since it is difficult to set the parameter values if the method is sensitive to the parameters in actual topic detection, the sensitivity of the method to the parameters is a critical factor. Fig. 4 compares the performance depending on the parameter selection. It shows the topic recall when ρ was varied to {0.005, 0.01, 0.015, 0.02, 0.025, 0.03, 0.035, 0.04} and ω was varied to {10, 30, 50, 70} and θ was varied to {100, 300, 500, 700, 900}. As shown in Fig. 4, topic recall for the FA dataset was stable when parameter value of ρ is above 0.025. When the ρ value is below 0.025, topic recall tends to increse for high ω values. For the ST dataset, topic recall was high when the θ value was in the range of 200 to 400. For the US dataset, the performance was the best when the ρ value was above 0.015 and the θ value was 300. It seems to be a little sensitive to the value of θ as the ρ value increases. Overall, for three data sets, topic recall was good when the ρ value was above 0.025, the θ value was in the range of 200 to 400, and the ω value was in the range of 70 to 90.

5.6. Evaluation of time required for topic detection

Since Twitter generates massive number of tweets within a short time period, it should be possible to detect topics quickly. Therefore, we evaluated the time required for topic detection using the compared methods. Fig. 5 shows the average time in seconds required for the ground-truth topic detections with the FA, ST, and

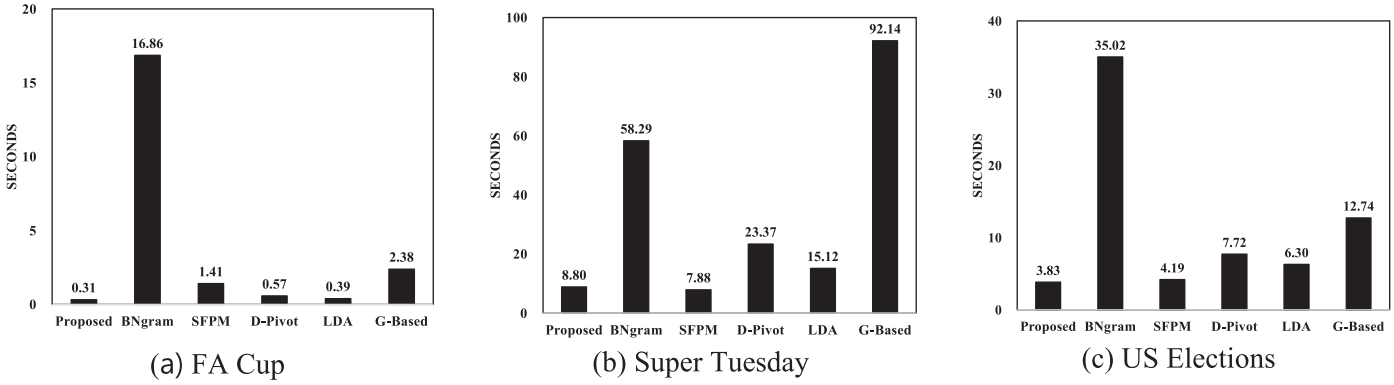


Fig. 5. The average time (in seconds) required by the compared methods on the three datasets.

US datasets. The computer used had a CPU Intel i7-3770(3.40 GHz), RAM 16GB and SSD Samsung 850 EVO 120GB.

The average time required for the ground-truth topic detections in the FA dataset was 0.31 s in the proposed method, indicating that the proposed method detected the topic most rapidly. In particular, the topic detection by the proposed method was about 50 times faster than that by BNgram which took the longest time. For the ST dataset, the topic detection was fastest in SFPM (7.88 s on average) among the compared methods, but the difference in time required by the proposed method was as small as about one second. For the US Elections dataset, the average time required for the topic detection was shortest in the proposed method (3.83 s), which showed a topic detection speed about 9 times faster than that of BNgram, which was the slowest among the compared methods.

For the three tested datasets, the proposed method and SFPM based on frequent pattern mining showed relatively short running time compared with other methods, while BNgram took longer overall for the computation of $df\text{-}idf_t$ scores of n-grams in time slots and hierarchical clustering of n-grams.

6. Conclusions

In this paper, we proposed a method for detecting topics from Twitter streams using HUPM. The proposed method includes a stage for calculating the utilities for words in each batch of tweets by the sliding window technique, a stage for determining the min-*util* on each batch, and a stage for extracting actual topic patterns from the candidate topic patterns using TP-Tree. We experimentally analyzed the topic detection performance of the proposed method in comparison with other methods. Notably, the proposed method showed a topic recall 5% higher than the other compared methods for the ST dataset, 6% higher for the US election data set, and 8% higher for the FA dataset. Regarding time spent for topic detection, the proposed method demonstrated short running time for the three datasets. Future studies may need to be conducted to apply the proposed method to other social media beside Twitter streams, to extend the scope and application of the topic detection method.

Author contribution section

- Hyeok-Jun Choi
  - Conceptualization; Data curation; Formal analysis; Methodology; Resources; Software; Validation; Roles/Writing - original draft;
- Cheong Hee Park
  - Conceptualization; Formal analysis; Funding acquisition; Methodology; Project administration; Supervision; Writing - review & editing.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF2015R1D1A1A01056622).

References

Aiello, L. M., Petkos, G., Martin, C., Corney, D., Papadopoulos, S., Skraba, R., et al. (2013). Sensing trending topics in twitter. *IEEE Transactions on Multimedia, 15*(6), 1268–1282.

Becker, H., Naaman, M., & Gravano, L. (2011). Beyond trending topics: Real-world event identification on twitter. In *Proceedings of the 5th international AAAI conference on weblogs and social media* (pp. 438–441).

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research, 3*, 993–1022 (3/1/2003).

Cormode, G., & Hadjieleftheriou, M. (2008). Finding frequent items in data streams. *Proceedings of the VLDB Endowment, 1*(2), 1530–1541.

Erra, U., Senatore, S., Minnella, F., & Caggianese, G. (2015). Approximate TF-IDF based on topic extraction from massive message stream using the GPU. *Information Sciences, 292*(C), 143–161.

Gaglio, S., Lo Re, G., & Morana, M. (2015). Real-time detection of twitter social events from the user's perspective. In *Proceeding of the IEEE international conference on communications* (pp. 1–6).

Fredkin, E. (1960). Trie memory. *Communications of the ACM, 3*(9), 490–499.

Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. In *Proceedings of the 2000 ACM SIGMOD international conference on management of data* (pp. 1–12).

He, Q., Chang, K., & Lim, E.-P. (2007). Analyzing feature trajectories for event detection. In *Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 207–214).

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).

Huang, J., Peng, M., & Wang, H. (2015). Topic detection from large scale of microblog stream with high utility pattern clustering. In *Proceedings of the 8th workshop on Ph.D. workshop in information and knowledge management* (pp. 3–10).

Ibrahim, M., Abdillah, O., Wicaksono, A. F., & Adriani, M. (2016). Buzzer detection and sentiment analysis for predicting presidential election results in a twitter nation. In *Proceedings of the 15th IEEE international conference on data mining workshop* (pp. 1348–1353).

Kim, H. D., Park, D. H., Lu, Y., & Zhai, C. X. (2012). Enriching text representation with frequent pattern mining for probabilistic topic modeling. In *Proceedings of the american society for information science and technology* (pp. 1–10).

Li, C., Sun, A., & Datta, A. (2012). Twevent: Segment-based event detection from tweets. In *Proceedings of the 21st ACM international conference on information and knowledge management* (pp. 155–164).

Liu, M., & Qu, J. (2012). Mining high utility itemsets without candidate generation. In *Proceedings of the 21st ACM international conference on information and knowledge management* (pp. 55–64).

Liu, Y., Liao, W., & Choudhary, A. (2005). A fast high utility itemsets mining algorithm. In *Proceedings of the 1st international workshop on utility-based data mining* (pp. 90–99).

Mathioudakis, M., & Koudas, N. (2010). TwitterMonitor: Trend detection over the twitter stream. In *Proceedings of the 2010 international conference on management of data* (pp. 1155–1158).

O'Connor, B., Krieger, M., & Ahn, D. (2010). TweetMotif: Exploratory search and topic summarization for twitter. In *Proceedings of the 4th international AAAI conference on weblogs and social media* (pp. 384–385).

Petkos, G., Papadopoulos, S., Aiello, L., Skraba, R., & Kompatsiaris, Y. (2014). A soft frequent pattern mining approach for textual topic detection. In *Proceedings of the 4th international conference on web intelligence, mining and semantics*.

Petrovic, S., Osborne, M., & Lavrenko, V. (2010). Streaming first story detection with application to twitter. In *Proceedings of the 2010 annual conference of the north american chapter of the association for computational linguistics* (pp. 181–189).

Phuvipadawat, S., & Murata, T. (2010). Breaking news detection and tracking in twitter. In *Proceedings - 2010 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology* (pp. 120–123).

Quercia, D., Askham, H., & Crowcroft, J. (2012). TweetLDA: Supervised topic classification and link prediction in twitter. In *Proceedings of the 3rd annual ACM web science conference* (pp. 247–250).

Ryang, H., & Yun, U. (2015). Top-k high utility pattern mining with effective threshold raising strategies. *Knowledge-Based Systems, 76*(1), 109–126.

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).

Sankaranarayanan, J., Samet, H., Teitler, B. E., Lieberman, M. D., & Sperling, J. (2009). TwitterStand: News in tweets. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems* (pp. 42–51).

Sayce, D. (2016). Number of tweets per day?. https://www.dsayce.com/social-media/tweets-day/. Accessed 15 November 2017.

Statista. Twitter: Number of monthly active users. (2017). https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/. Accessed 15 November 2017.

Teh, Y., Newman, D., & Welling, M. (2007). A collapsed variational bayesian inference algorithm for latent dirichlet allocation. In *Proceedings of the 19th international conference on neural information processing systems* (pp. 1353–1360).

Tseng, V. S., Wu, C. W., Fournier-Viger, P., & Yu, P. S. (2016). Efficient algorithms for mining top-K high utility itemsets. *IEEE Transactions on Knowledge and Data Engineering, 28*(1), 54–67.

Wang, C., Zhang, M., Ru, L., & Ma, S. (2008). Automatic online news topic ranking using media focus and user attention based on aging theory. In *Proceeding of the 17th ACM conference on information and knowledge management* (pp. 1033–1042).

Weng, J., & Lee, B. (2011). Event detection in twitter. In *Proceedings of the 5th international AAAI conference on weblogs and social media* (pp. 401–408).

Wu, C. W., Shie, B.-E., Tseng, V. S., & Yu, P. S. (2012). Mining top-K high utility itemsets. In *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 78–86).

Zhang, W., Yoshida, T., Tang, X., & Wang, Q. (2010). Text clustering using frequent itemsets. *Knowledge-Based Systems, 23*(5), 379–388.

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