International Journal of Information Technology & Decision Making

EWNSTREAM+: EFFECTIVE AND REAL-TIME CLUSTERING OF SHORT TEXT STREAMS USING EVOLUTIONARY WORD RELATION NETWORK

SHUIQIAO YANG

Data Science Institute, University of Technology Sydney, Sydney, Australia.
shuiqiao.yang@uts.edu.au

GUANGYAN HUANG*

School of Information Technology, Deakin University, Burwood, Victoria, 3125, Australia.
guangyan.huang@deakin.edu.au

XIANGMIN ZHOU

School of Computer Science and Information Technology, RMIT University, Melbourne, Victoria, 3000, Australia.
xiangmin.zhou@rmit.edu.au

VICKY MAK

School of Information Technology, Deakin University, Burwood, Victoria, 3125, Australia.
vicky.mak@deakin.edu.au

JOHN YEARWOOD

School of Information Technology, Deakin University, Burwood, Victoria, 3125, Australia.
john.yearwood@deakin.edu.au

The real-time clustering of short text streams has various applications, such as event tracking, text summarization and sentimental analysis. However, accurately and efficiently clustering short text streams is challenging due to the sparsity problem (i.e, the limited information comprised in a single short text document leads to high-dimensional and sparse vectors when we represent short texts using traditional vector space models), topic drift and the fast generated text streams. In this paper, we provide an effective and real-time Evolutionary Word relation Network for short text Streams clustering (EWNStream+) method. The EWNStream+ method constructs a bi-weighted word relation network using the aggregated term frequencies and term co-occurrence statistics at corpus level to overcome the sparsity problem and topic drift of short texts. Better still, as the query window in the stream shifts to the newly arriving data, EWNStream+ is capable of incrementally updating the word relation network by incorporating new word statistics and decaying the old ones to naturally capture the underlying topic drift in the data streams and reduce the size of the network. The experimental results on a real-world dataset show that EWNStream+ can achieve better clustering accuracy and time efficiency than several counterpart methods.

*corresponding author

Keywords: short text stream; clustering; topic discovery; event detection.

1. Introduction

1.1. *Motivation*

With the rapid development of social networks such as Twitter, Quora, and Yelp, tremendous amounts of text data are continuously generated by online users, usually in the form of short texts (e.g., tweets, questions and reviews). Real-time clustering analysis is an important unsupervised technique to automatically extract the important knowledge from the massive short text streams. The real-time clustering for short text streams could be used in many different applications, such as event detection and prediction [1, 2], sentimental analysis [3–6] and text summarization [7, 8]. For example, it is promising to mine useful information from social media published by users for smart city development [9–11]. The users are recognized as "social sensors" [12, 13] and the data generated by them are closely related to different aspects of cities. Organizing the semantically similar text data posted by the residents into concise clusters in real time, we can detect and predict the possible social activities and events (e.g., crime events, financial risks, traffic congestion, energy consumption and pollution) [1, 14–19]. Such real-time analysis on text data is expected to be much efficient and more comprehensive than using the traditional questionnaire and thus allows for timely response to emergent events or disasters [12, 20].

Fig. 1 shows an example of the clustering processes for short texts in streams. When social events like sports or disasters appear, lots of social media users would make timely post about the events that they have experienced. Thus, lots of texts would appear in the form of data streams. For example, when the *swine flu* happened in 2009, lots of people were affected and made posts on social media which led to the related data streams shown in the left part of Fig.1. It becomes meaningful to group similar posts into the same text cluster, so that important information about the events could be discovered to support the related decision makings. With the continuous generation of short texts from social networks, the clustering system needs to quickly determine the cluster membership of the arriving short texts and assign the data into different groups (shown as the right part of Fig. 1). Based on the semantically and logically well organized short text groups, various applications, such as topic summarization and event detection, can be realized. As the short texts are closely related to the real-world events and the changes of content are unforeseeable in the stream [21], the clusters (shown as the right part of Fig. 1) should be dynamically generated to cope with the topic drift in the stream.

1.2. *Challenges*

Our goal is to achieve real-time and effective short text clustering from the data streams. Despite its significant importance in various practical applications, the task is nontrivial because of the following challenges:

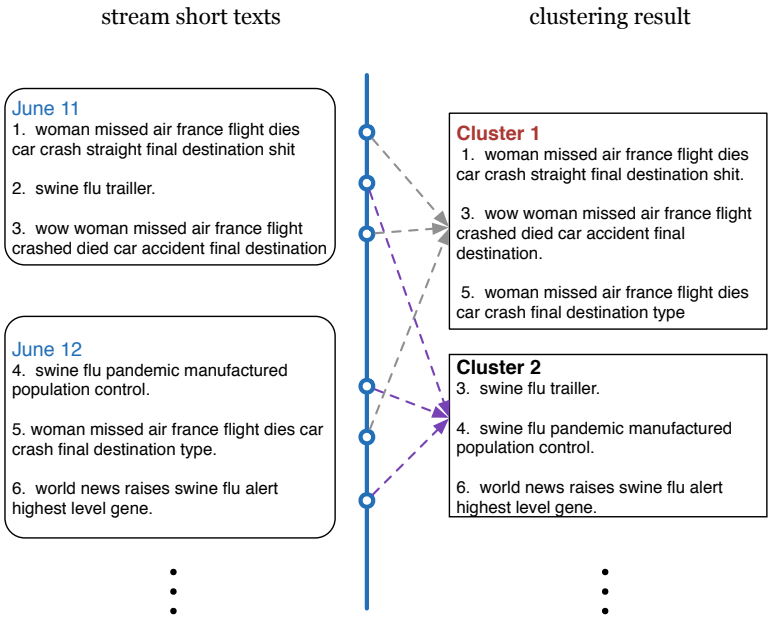

Fig. 1: Illustration of stream short text clustering.

- Extracting effective semantics from the short text data is difficult due to the short text sparsity problem. Unlike traditional documents, short texts such as tweets are generally sparse and contain fewer words. For example, the length of a tweet is limited to 140 characters. Therefore, the sparse semantics, less contextual information and limited word statistical patterns are included in short texts. Compared with traditional documents, it is much more difficult to extract effective semantics from short text documents.
- It is hard to capture the topic drift in the short text streams. The users of social media are posited as social sensors and monitor the real world with instant short text posts that are closely related to unforeseeable events. Thus, topics keep fading and thriving in the data streams. Also, the feature space (i.e., terms) that supports different topics is also evolving and cannot be assumed to be static. Therefore, it is challenging to effectively capture the topic drift in the text streams.
- The dynamically accumulated text stream data make real-time clustering problematic. When an emergent event outbreaks, it is necessary to cluster the related text data quickly for instant analysis and action. However, with a large number of active users in social networks, the generated data from users could be uncomfortably large. Thus, the requirement of real-time clustering becomes another challenge in our task.

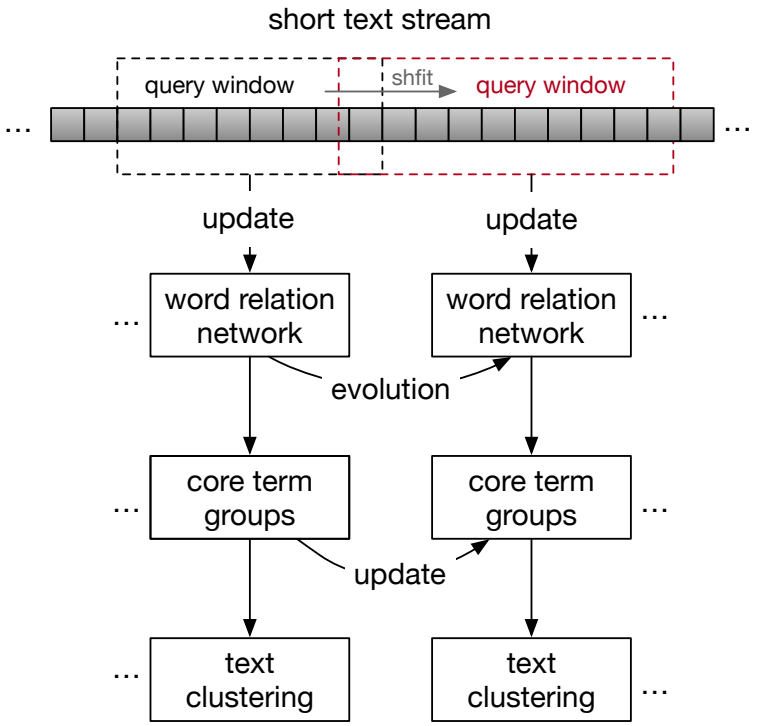

Fig. 2: Illustration of the proposed method EWNStream+.

Generally, the existing work for real-time short text clustering can be classified into two categories: the dynamic cluster structure maintenance related methods [7, 10, 22–26] and topic evolution tracking related methods [27–33]. The dynamic cluster structure maintenance approaches consider each document as a basic unit and maintain a list of cluster structures in an online fashion to capture the evolution of data in the stream. Short texts are usually represented as term-weighted vectors [7] and they are either merged into an existing cluster or form a new one based on a predefined similarity threshold. But these methods overlook the negative effects caused by the inaccurate similarity measurement due to the high dimensional and sparse vector representations for short texts [34].

The topic evolution tracking related methods consider inferring the latent topics that govern the generation of short texts in a dynamic way. To determine the latent topic index for each short text, techniques, such as probabilistic generative model and Gibbs sampling [35], are adopted. However, to get the posterior topic distributions, multiple iterations of sampling operations on the documents [27, 29] are required in those methods, which are time-consuming and inefficient, and thus preventing the real-time processing for large scale streaming text data.

1.3. *Contributions*

In this study, we propose an effective and real-time short text clustering method named EWNStream+. Our insight behind the design of EWNStream+ is that short texts generally contain few meaningful keywords and the topic of a short text cluster can be summarized with several core terms. Therefore, it is promising to extract the core terms as a virtual cluster center to attract the semantically similar short texts into a cluster. However, identifying different core term groups for various short text clusters in the text stream is nontrivial. Firstly, the processes to identify the core term groups must be designed in an unsupervised way since there is no available prior knowledge to differentiate between the core terms and the normal terms. It is also unrealistic to manually label the large corpus data. Secondly, the relations between core terms are dynamically changing due to the unforeseeable topic drift in the streams. One of the distinguished attributes of short texts is its dynamic nature. When events appear, evolve and fade, the changes in the corresponding core term groups also need to be addressed.

Motivated by the above observations, EWNStream+ exploits the aggregated word patterns from short texts in the query window into a bi-weighted word relation network and discovers all the potential core term groups using a novel fast searching method. The core term groups can be regarded as different communities on the word relation network [8, 36]. Then, an efficient and real-time short text clustering strategy is designed to absorb short texts that are semantically similar to the virtual cluster centers supported by different core term groups. The bi-weighted word relation graph is a node-weighted and edge-weighted word relation graph constructed by exploring the corpus-level word frequency and word co-occurrence frequency. As mentioned before, the sparsity of short text leads to difficulties in mining useful semantics at the document level. Most of existing methods take each short text as a basic unit to measure the similarity at the document level. On the contrary, we exploit the word relation patterns at the corpus level. With the aggregated word statistical patterns, we can compress the latent topic information of short texts into the bi-weighted word relation graph and thus overcome the sparseness of short texts.

To find the core term groups efficiently, we develop a fast core term group searching (FCTGS) method on the bi-weighted word relation graph. FCTGS is designed to discover the core terms that are closely bonded to each other as a group. For the core terms in a group, they are recognized not only as important terms but also required to be close to each other. To achieve this purpose, FCTGS exploits a new closeness metric between nodes and uses a highly weighted node to search its core term partners. To group short texts, the discovered core term groups are used as the virtual cluster centers. Short texts are clustered by choosing their closest centers based on a defined semantic similarity metric.

To cope with the topic drift in short text streams, the bi-weighted word relation network is designed to be able to evolve with the new arriving of text data. When

the query window shifts, the bi-weighted word relation graph will be updated by incorporating the new word relations. Meanwhile, as the real world events can not last forever, it is necessary to gradually filter out the old topics. A decay strategy is introduced on the bi-weighted word relation network to naturally reduce the outdated word relations and thus to ensure the time efficiency of EWNStream+. We also maintain a list of cluster abstract structures that include fields, such as the temporal statistics, the core terms and the cluster size. We use temporal statistics from the cluster abstract to infer if a cluster is active or not. Fig.2 illustrates the framework of EWNStream+. We process the short texts in a query window from the stream. The word relation network is updated when the query window shifts as new data arrive in the stream. Meanwhile, the core term groups are updated from the word relation network to capture the change and cluster the new data.

The proposed EWNStream+ is an extension of our previous EWNStream [37]. Compared with EWNStream, the proposed EWNStream+ improves the core term group searching by introducing a new closeness metric on the bi-weighted word relation network to comprehensively search the core terms as a group. In addition, EWNStream+ incorporates a decay policy on the bi-weighted word relation network to filter the old word statistics and thus aligns the fading of topics in the real world. The major advantages of EWNStream+ over its preliminary version are two-fold: (1) EWNStream+ provides a novel search method to find the core term groups on bi-weighted word relation network; and (2) EWNStream+ introduces a novel decay strategy to naturally reduce the outdated word relations and prevents the graph from oversize. Our experiments prove that the new core term group searching method and the decay policy are useful for improving the clustering accuracy.

Compared with existing methods, the proposed EWNStream+ exploits the aggregated term co-occurrences and term frequencies into a node-weighted and edge-weighted word relation network as the basis to discover the potential clusters. The terms' statistics are compressed into the word relation network to discover core term groups, which have demonstrated to be useful in summarizing short text clusters [8,38]. The proposed FCTGS strategy works efficiently on the bi-weighted network to discover core term groups. To meet with the dynamic changing attribute of text data in streams, the bi-weighted word relation network in EWNStream+ is designed to be able to naturally evolve by incorporating the new term statistics from the text streams and thus capture the new information. Furthermore, EWNStream+ does not require the representation of short text for distance measurement and only need one time access for the text data in the streams. Hence, EWNStream+ is more efficient than the existing methods. We conduct extensive experiments on a real-world Twitter dataset, which contains around 140,000 tweets. The experimental results demonstrate that the proposed method achieves much better clustering quality with the Normalized Mutual Information accuracy reaches around 86% and this is significantly higher than its counterpart methods. Meanwhile, EWNStream+ can be maximum of 30 times faster than dynamic topic model (DTM) method [27]. The discovered core term groups and the short text clusters are also meaningful to

reflect the real-world events.

Thus, the main contributions of this work are summarized as follows:

- We propose EWNStream+ for effective and real-time short text stream clustering. EWNStream+ overcomes the short text sparsity issue by aggregating the term relation and term frequency statistics in a node-edge weighted word relation network. The effectiveness of EWNStream+ is underpinned by its abilities in accurately and dynamically determining the cluster memberships of short texts with the identification of the core term groups for clusters.
- In EWNStream+, we design a fast core term group searching (FCTGS) strategy to effectively search the core term groups. FCTGS searches the core terms in a weight descending order to quickly find core terms that meet a new closeness metric between nodes. Also, FCTGS finds the most important terms in the large network to represent short text clusters. Thus it enables our proposed method to perform significantly more efficient than the other methods based on the probabilistic topic modelling.
- In EWNStream+, we handle the dynamic change of short text streams to overcome topic drift by designing an evolutionary scheme of the word relation network, which incrementally incorporates new word statistics into the word relation network. Meanwhile, we also introduce a decay strategy to filter outdated word relations.
- The experimental results on a real-world Twitter dataset show that EWNStream+ can achieve better clustering accuracy and time efficiency than several counterpart methods.

The rest of this paper is organized as follows. Section 2 presents related work. Section 3 introduces preliminaries. Section 4 details the proposed approach. Experimental results are reported in Section 5. Discussion is presented in Section 6. Conclusion is made in Section 7.

2. Related Work

In this section, we review the state-of-the-art for short text stream clustering methods. To the best of our knowledge, the current literature for short text stream clustering can be generally classified into two categories: dynamic cluster structure maintenance and topic evolution tracking. In the following sections, we analyze the related studies in the two categories and discuss their advantages and disadvantages in detail.

2.1. *Methods for dynamic cluster structure maintenance*

This category of methods group streaming text into clusters based on the predefined similarity or distance metrics between texts. To capture the dynamics in the data

stream, a series of active clusters are maintained with well-defined cluster structures to discard the old data or absorb new data into the current clusters during the clustering process [39]. The text data are normally represented as feature vectors (i.e., TF-IDF vectors) and similarity metrics, such as cosine similarity and Euclidean distance, are adopted.

One of the classic stream clustering frameworks that can be used in text stream clustering is CluStream [22], which adopts a cluster feature vector named *micro cluster* to summarize the statistical information for a cluster and to decide the cluster membership for the new data object. Each micro cluster contains five components, such as the number of the current data objects, the linear sum of the data objects vectors, the sum of the data objects arrival time, etc. For each new data object, it will be absorbed into its closest cluster according to the Euclidean distance between the new coming data object and the corresponding micro cluster. CluStream maintains a list of cluster feature vectors to capture the evolution of stream data. Similarly, ClusTree has been proposed [40] to incrementally update a tree structure for stream data clustering. ClusTree uses micro cluster to compact the representations for the data distribution. The micro clusters are dynamically maintained for recording the changing of the data stream by updating the mean and variance of micro clusters. DenSream [23] has been proposed to cluster stream data based on the data densities. DenStream also uses a cluster feature vector for representing clusters. Specifically, DenStream adopts three cluster structures: *core micro cluster*, *potential core micro cluster* and *outlier micro cluster* for stream data clustering. The *core micro cluster* is a dense micro cluster data structure that includes data objects that are close to each other according to the predefined distance threshold. The *potential core micro cluster* represents potential clusters that may become *core micro cluster*. The *outlier micro cluster* is used for outlier detection.

The spherical k-means algorithm has been extended into [24] an online version and proposed online spherical k-means (OSKM) based on Winner-Take-All competitive strategy. In OSKM, the centroid of a cluster will be dynamically updated if new text data are absorbed in the cluster. The updates for the centroid follow the gradient decreasing direction. The appearance of new clusters is also based on the similarity between a new text and the current cluster centroids. If the distance is above a threshold, then a new cluster centroid will be created. A lexical clustering model has been built [25] for short text stream clustering using the frequent word pairs. A fraction of texts from each batch of data streams is first grouped into a cluster based on the word pairs frequency. If the frequency of word pairs is greater than the mean frequency value calculated from all the word pairs, then the word pairs are used to represent the clusters and the texts that contain the word pairs will be formed as clusters. To maintain the clusters dynamically, the proposed lexical clustering model relies on a cluster feature structure to represent the active short text groups during the data stream clustering process. Specifically, they construct the cluster vector with frequencies in texts, number of texts and number of words. Then, based on the cluster structure, the assigned probability for the rest texts to

each cluster will be calculated. A text stream clustering method has been proposed in [26] for crisis management using social media text data streams. In this method, TextClust [41] is adopted, which contains a two-phase micro and macro clustering. The micro clustering phase incrementally forms micro clusters to represent the topics in the text stream. The macro clustering phase group text data in the specified time interval based on the cluster centers formed in the first phase.

Sumblr has been proposed [7] to cluster and summarize the large-scale tweets stream. Sumblr contains two core modules: online incremental clustering for streaming tweets and historical tweets summarization. In the clustering part, tweet cluster vectors (TCV) were used to store the statistical features of tweet clusters. TCV contains the statistical information of the current cluster, such as the sum of normalized vectors, the sum of weighted textual vectors, the sum of time-stamps and the number of tweets. For each new arriving tweet, the similarity between a new tweet and the current clusters will be computed. Based on the similarity, the new tweet is merged into the current cluster or form a new cluster. Sumblr also contains a summarization module that compresses a large number of tweets in a cluster with a pyramidal time frame to store snapshots of the TCV at different levels of granularity depending on the current time. StreamCube [10] have been proposed which focuses on the clustering of *hashtag* streams from Twitter for event detection. In StreamCube, the spatial and temporal aspects are considered to detect the bursty hashtags. The hashtags are represented by its co-occurenced words in a stream. To group the similar hashtags in tweets stream, the similarity between hashtags is computed and compared with the current event clusters. Similar hashtags in a group are used to represent a spatial-temporal event.

2.2. *Methods for topic evolution tracking*

Unlike methods for the cluster structure maintenance, which need to maintain a list of cluster structures and absorb texts into different clusters based on a similarity threshold, the topic methods for evolution tracking determine the cluster memberships for the data objects through inferring and tracking the time-varying topics and cluster texts that share the same topics.

Many of those methods extend the classic Latent Dirichlet Allocation (LDA) [35] model, which is used for static corpus topic learning, into the streaming corpus topic learning and tracking. For example, dynamic topic model (DTM) [27] has been proposed for discovering topics from streaming corpus. The dynamic topic model consists of multiple Latent Dirichlet Allocation (LDA) units. Each LDA unit learns the topic distribution for documents in specified time spans. The prior parameter setting for one LDA model in DTM is based on the posterior parameters that learned by its previous LDA models. In this way, the topics learned at time slice t is smoothly evolved from the previously learned topics at time slice $t-1$. Therefore, the inferred topics at different time periods can evolve more naturally.

Hence, the cluster membership can be determined based on its topic distributions. Similarly, the streaming-LDA [28] has been proposed to model the word and topic dependencies between the consecutive documents in a data stream. The dependency of topic distributions for the adjacent documents is assumed to follow a Dirichlet distribution governed by the hyperparameters.

A topic tracking model (TTM) [42] has been proposed for analyzing the time-varying consumer purchase behavior. Similar to DTM, TTM also consists of several basic LDA units that used to mine users' purchase interest in different time slots. The difference between TTM and DTM is that the LDA units in TTM also have long prior parameter dependence, which is supposed to capture the topic evolution in the long-time period. A dynamic clustering topic model (DCT) [29] has been proposed to track the time-varying distributions of topics for short text streams. The major difference between DCT and TTM is that DCT adopts the Dirichlet mixture model (DMM) as the basic topic discovery unit. DCT models the temporal dynamics of stream texts using multiple Dirichlet mixture models for the different time periods of the stream texts. The Dirichlet mixture model infers the latent topics for short text corpus with the assumption that each short text is generated from a single latent topic. To smoothly learn the topic distributions, DCT adopts similar strategies with DTM [27] and TTM [42] that the prior parameters for DMMs have short or long term dependency with each other. The inferred latent topic indexes are used as cluster indexes for the short texts. However, one of the drawbacks for DTM, TTM and DCT is that they can not detect new topics due to the fixed topic number settings. Recently, MStream [30] has been proposed based on the Dirichlet process multinomial mixture model. Compared with DCT, MStream exploits the Dirichlet process on DMM to capture the dynamic appearance of new topics. In MStream, the Dirichlet process mechanism allows the new arriving short texts to choose a current topic (or cluster) or a new topic based on the topic-document probability. However, both DCT and MStream are probabilistic generative models, which need multiple sampling operations on each document to get the topic distribution.

The traditional topic model has been combined with artificial neural network [33] for the topic evolution detection in text streams. They have incorporated a context-aware topic layer and a Long Short Term Memory (LSTM) layer into the traditional Recurrent Chinese Restaurant Process (RCRP) for mining the evolutionary trends of the streaming text. The purpose of the adoption for the context-aware topic layer is to capture the global context-aware semantic coherences while the LSTM layer is exploited to learn the local dynamics and semantic dependencies from the evolutionary Chinese Restaurant Process. Even though the proposed method shows better accuracy, however the model needs to be trained before used into the streaming text topic detection. Therefore, it may be time-consuming and not suitable to the large scale short text streaming data. Similarly, The topic evolutionary have also been proposed to mine from short text streams with a nonparametric model (NPMM) [32] which exploits a pre-trained word embeddings as knowledge base to infer the topic number from the data streams. Considering the sparsity issue buried

in short text data, they proposed to employ a spike and slab function to alleviate the potential sparse features of the inferred topic-word distributions in stream short text mining. The word embedding tend to be trained with large and static corpora, thus the potential drawback of the proposed NPMM method is the inflexibility of relying on the auxiliary word embedding to enhance the short text features.

Now we discuss the advantages and disadvantages of the two categories of short text stream clustering methods. The advantages of the first category (i.e., methods for cluster structure maintenance) are their straightforwardness and easier implementation. With the dynamically updated cluster structures, the new information or concept could be absorbed into the active clusters and thus allowing for distance measurement between the new texts and the up to date cluster centers. However, the drawback is that the document is usually represented as high dimensional vectors which lead to the inaccuracy of measuring the distance between texts and cluster centers. Hence, this type of methods may be improved with the adoption of sophisticated feature selection strategies [43–49]. For instance, an unsupervised learning technique [46] is used to select the key features to improve the accuracy of the text clustering. Although the adopted feature selection strategies may obtain accurate clusters, they may not achieve real-time performance and be scalable for dynamically processing text streams.

As for the second category (i.e., topic evolution tracking), their advantages lie in that they can discover deeper features from the text streams for short text clustering by mining high-level topic information. For example, the dynamic topic model (DTM) [27] exploits the subtle word co-occurrence patterns to infer if the words could belong to the same topic or not. However, the disadvantage of these methods is their the higher time complexity. In order to infer the topic-document or topic-word distributions, these methods need to sample each word from each document multiple times to get the stable status for the probability distributions.

3. Preliminaries

In this section, we first formulate the problem for short text stream clustering. Then, we introduce a data structure used in EWNStream+ for managing the short text clusters called *Cluster Abstract* (CA).

3.1. *Problem modeling*

A short text stream denoted by $S_{\leq t} = \{\cdots, d_{t-2}, d_{t-1}, d_t\}$ is a series of timestamped documents that arrive in a chronologically order, where t denotes the generation time of the latest short text in $S_{\leq t}$. The problem of short text stream clustering can be converted to the problem of designing an algorithm f that essentially satisfies:

$$S_{\leq t} = \{\cdots, d_{t-2}, d_{t-1}, d_t\} \xrightarrow{f} \mathcal{U}_{\leq t} = \{C_1, C_2, \cdots, C_z\}, \tag{1}$$

where $\mathcal{U}_{\leq t}$ represents the clustering results that include a set of z clusters for the short texts in stream $S_{\leq t}$ and each cluster C_i is a collection of semantically similar

short texts. Note that, $S_{\leq t}$ will be updated as $S_{\leq t'}$, when the time t increases to t' due to the shifting of the text window and the new data are denoted as $S_{t \to t'} = \{d_t, \cdots, d_{t'}\}$. Each short text document, d, contains a bag of words. To ensure the text quality, we perform the general text cleaning steps, such as removing stop words, removing non-English characters and stemming.

3.2. *Text cluster structure*

During the short text stream clustering processes, it is necessary to maintain a list of active cluster data structures for dynamical cluster membership prediction. Here, we define a new data structure called cluster abstract (CA), which maintains the key information for a short text cluster. CA is a variant of the cluster structure defined in the Sumblr [7] and CluStream [22]. Compared with the previous ones, CA provides a new field called *pd*, which is a core term set used as a virtual cluster center to absorb similar short texts.

Definition 1 (Cluster Abstract). For a short text cluster C that contains a list of short texts, its CA is defined as a tuple: $CA(C) = (n, st_1, st_2, pd, I)$, where

- $n = |C|$ is the number of short texts in cluster C,
- $st_1 = \sum_{i=1}^{n} ts_i$ is the sum of timestamps of each $d_i \in C$,
- $st_2 = \sum_{i=1}^{n} ts_i^2$ is the squared sum of timestamps of each $d_i \in C$,
- $pd = (k_1, k_2, \ldots, k_m)$ is a core term set,
- I is the index number of the cluster C.

CA provides a concise summarization for a short text cluster. Its field, *pd*, is used as a virtual cluster center to absorb semantically similar texts. The fields, st_1 and st_2, summarize the data arriving patterns and are used to determine the momentum of the corresponding cluster or topic.

4. The EWNStream+ framework

In this section, we detail the EWNStream+ framework. We first introduce the concept of bi-weighted word relation network and propose a fast core term group searching (FCTGS) strategy based on the word relation network. We then explain how to update the word relation network with a decay model and cluster the short texts in streams based on the dynamically searched core term groups.

4.1. *Bi-weighted word relation network*

Due to the limited length, short texts are quite sparse and lack of context. Representing short texts as term weighted vectors leads to the high-dimensional and sparse issues, and thus may cause undiscriminating similarity measurement between short texts. Therefore, we are motivated to exploit corpus-level word patterns to overcome the document-level sparsity. Here, we exploit the aggregated word patterns from multiple short texts in the query window of the data stream to construct the bi-weighted word relation network. For a given batch of short texts,

$S_{t_s \to t_e} = \{d_{t_s}, \cdots, d_{t_e}\}$, in query window, $Q = \{t_s, t_e\}$, where t_s and t_e represent the start and the end of the batch $S_{t_s \to t_e}$, the bi-weighted word relation network, $G = \{V, E\}$, is constructed as follows:

- Each node, $u \in V$, is weighted by its corpus-level term frequency in Q, denoted as $g(u) = \sum_{d \in S_{t_s \to t_e}} I(u \in d)$,
- Each edge, $e_{u,v} \in E$, is weighted by its corpus-level term co-occurrence frequency between the two nodes u and v in Q, denoted as $g(u, v) = \sum_{d \in S_{t_s \to t_e}} I(u \in d \; and \; v \in d)$,

where I is an indicator function. For each short text, $d_i \in S_{t_s \to t_e}$, it can be presented as a bag of words denoted as $d_i = \{w_{i,1}, \cdots, w_{i,n_i}\}$, where $w_{i,j}$ is the j-th word (or term) in d_i and n_i is the number of words in d_i. Due to the limited length and sparsity of short texts, many words may have low co-occurrence frequencies at the corpus level in a given query window. To diminish these unimportant co-occurrences and reduce the size of the word relation network, we only retain the edge $e_{u,v}$ between two nodes u and v if the edge weight, $g(u, v)$, is greater than a predefined threshold, γ. We set γ as 30 in our experiments.

After constructing the bi-weighted word relation network using the short texts in the initial query window of the data stream, we can start to mine core term groups for different short text clusters. Note that the word relation network is updated incrementally when the query window shifts forward and new short texts flow into the stream.

4.2. *Fast core term group searching*

Now, we introduce an effective strategy to discover core term groups for representing different short text clusters from the word relation network. The discovering of core term groups is non-trivial. The difficulties lie in that core terms for different clusters may have connections to each other and it is hard to find the dividing boundaries in the word relation network. According to our observations as illustrated in Fig. 3, the core terms for a short text cluster (or a topic) show two patterns: (1) the node weights of core terms in a cluster are significantly higher than all the other terms in a cluster; (2) the edge weights among core terms in the same group are significantly greater than the core terms in different groups. That is, the core terms for a short text cluster (or a topic) are close to each other and have relatively higher term frequency. From Fig. 3, we can see that the core terms from one group tend to have strong relation (i.e., edge weight) than terms across different groups. Meanwhile, the core terms of a group tend to have higher node weight than the normal terms (e.g., the different node weight between terms **drive** and **air**. Therefore, we are motivated to locate a core term group that meets the following two requirements: (1) for each core term in the group, it should have relatively higher node weight; (2) for all the core terms in the group, they should have relatively higher edge weight.

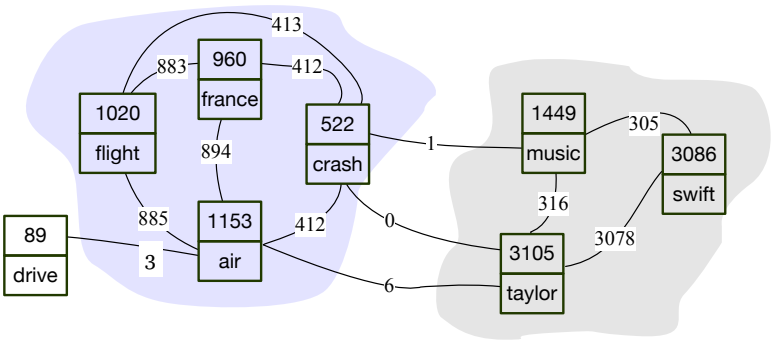

Fig. 3: Illustration of a bi-weighted word relation network. The node weight is calculated using the corpus-level term frequency and the edge weight is calculated using the corpus-level term co-occurrence frequency inside a query window.

To satisfy the goals, we first define the closeness between two directly connected nodes/terms in the word relation network. The closeness between the node u and its neighbor node v, is defined as follows:

$$C(u,v) = \begin{cases} \alpha \frac{N(u) \cap N(v)}{N(u) \cup N(v)} + (1-\alpha) \frac{g(u,v)}{g(u)} & \text{if } g(u) > g(v) \\ \\ \alpha \frac{N(u) \cap N(v)}{N(u) \cup N(v)} + (1-\alpha) \frac{g(u,v)}{g(v)} & \text{if } g(u) < g(v), \end{cases} \tag{2}$$

where $\alpha \in [0,1]$ is a weight coefficient, $N(u)$ and $N(v)$ are the neighboring nodes set for u and v, $g(u,v)$ indicates the edge weight between u and v, $g(u)$ and $g(v)$ are the node weight for u and v. Here, the closeness between the two nodes, u and v, are considered from two aspects. The first part of $C(u,v)$ indicates the Jaccard similarity between u and v with the consideration of the common co-occurred terms. The second part of $C(u,v)$ indicates the co-occurrence similarity between u and v with the consideration of the co-occurrence proportion rate in regards to the node weight. The closeness metric is symmetric and its value range is in [0, 1].

To locate a core term group, we design an approximate method for fast searching by choosing a node that has higher node weight than its neighbors and absorbing the other core terms from its neighbors based on the closeness metric defined in Eq. 2. Fig 4 illustrates the main idea of FCTGS. we first find node u that has the highest node weight than its neighboring nodes in the word relation network. If the closeness, $C(u,v)$, between u and its neighbor v is greater than a predefined threshold β, then the two nodes u and v formulate a potential core term group. We search all the neighbor nodes of u to find all the core terms surrounding to u. As node u has the locally highest frequency, the selected neighbor nodes of u also show a relatively higher frequency in order to meet the closeness requirement. Therefore, u and the other core term nodes are closely related to each other and have a relatively higher frequency at the same time. We denote the core term groups as $K = \{k_1, \cdots, k_m\}$, where m is the discovered term group number and the number

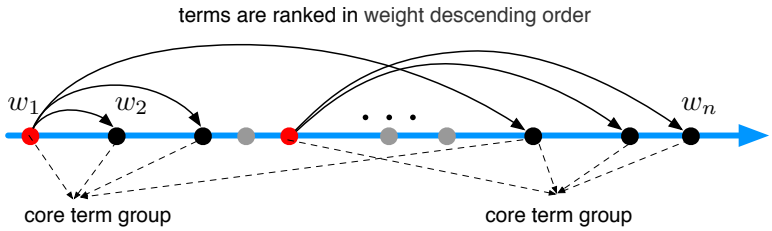

Fig. 4: Illustration of FCTGS for fast core term group searching. The terms in graph G are searched in a node weight descending order. The red terms indicate higher node weight that are used to absorb the other core terms that meet the closeness requirement.

Algorithm 1: Fast Core Term Group Searching (FCTGS).

Input: word graph $G = (V, E)$, α, β
Output: core term group list K

```
1  Get node list from G in descending order as L
2  Initialize P = 0 and K = ∅
3  for u ∈ L do
4      if P[u] ≠ 1 then
5          Initialize k_u = ∅
6          for v ∈ N(u) and P[v] ≠ 1 do
7              if C(u, v) ≥ β then
8                  k_u.append(v)
9                  P[v] ← 1
10             end
11         end
12         if k_u != ∅ then
13             K.append(k_u)
14         end
15     end
16 end
```

of term groups may change when the new short text flows in. The i-th core term group is denoted as $k_i = \{w_{i,1}, \cdots, w_{i,|k_i|}\}$, where $|k_i|$ represents the number of core terms in k_i.

Algorithm 1 shows the processes to discover core term groups with word relation network, G, as input. We first get the node list from G in descending order, initialize an array variable, $P = 0$, as the processed flag for each node and an empty set K for retaining all the core term groups at lines 1-2. Sorting all the nodes in descending order at line 1 is to check all the nodes on the word relation network in a fixed order.

Then, we use a loop to search the core term groups at lines 3-16. Since we process the nodes in descending order, the node u at line 5 naturally has a larger node weight than its neighbors that excluding the nodes that been marked as processed. Then, we absorb the nodes that meet the closeness requirement at lines 6-9. Since some detected nodes may have no partner nodes to form the core term group, we only retain the non-empty keyword groups at lines 12-13.

4.3. *The decay model*

For most topics (e.g., emergent news, sports finals and concerts) in the short text streams (e.g., tweet streams), they usually last for a short period. Therefore, the importance or freshness of topics should be decayed with time. A popular solution is the exponential time decay function [40], which has the general form shown as follows:

$$f(t') = \alpha^{-\lambda(t'-t)}, \tag{3}$$

where λ and α control the decay rate. λ is greater than 0 and higher value indicates a more rapidly vanishing rate. α is greater than 1 to ensure that $f(t')$ is monotonous decreasing. We set α as constant e and λ as 0.2 in our method. Unlike the other data type represented by real values, the topics of short text data are underpinned by several core terms. In order to fade the outdated topics, we decay the node weight and edge weight (word frequency and word co-occurrence frequency) in word relation network, so that topics supported by specified word relation patterns (i.e., word closeness) can not be searched out using FCTGS in Algorithm 1.

4.4. *The evolutionary updating for the word relation network*

Instead of constructing the word relation network from scratch when the query window moves forward, we update it gradually when new text data flow in. We assume that the short text stream before time t is denoted as $S_{\leq t}$ and its corresponding word relation network is denoted as $G_t(V, E)$. When the query window moves forward from time t to time t', a batch of new short texts denoted as $S_{t\to t'}$ would flow into the stream.

The bi-weighted word relation network is designed to dynamically evolve from $G_t = (V, E)$ at time t into $G_{t'} = (V', E')$ at time t' by updating the node weight and edge weight. The updating operations for G consist of two types: decaying the previous word statistics in the stream before time t and incorporating new word statistics from texts in $S_{t\to t'}$. The weight of a node u in $G_{t'}$ is updated as

$$g_{t'}(u) = \begin{cases} \alpha^{-\lambda(t'_u - t_u)} g_t(u) + \sum\limits_{d \in S_{t\to t'}} I(u \in d) & \text{if } u \in S_{\leq t} \text{ and } u \in S_{t\to t'} \\ \alpha^{-\lambda(t'_u - t_u)} g_t(u) & \text{if } u \in S_{\leq t} \text{ and } u \notin S_{t\to t'} \\ \sum\limits_{d \in S_{t\to t'}} I(u \in d) & \text{if } u \notin S_{\leq t} \text{ and } u \in S_{t\to t'}, \end{cases} \tag{4}$$

where t_u and t'_u indicate the last appearance time of node u in $S_{\leq t}$ and $S_{t\to t'}$, respectively. Eq. 4 indicates that the weight of a node u can be updated based on the term statistics of u in $S_{\leq t}$ and $S_{t\to t'}$. If the node u has appeared in both streams, then we need to decay its previous weight and incorporate new weight. If node u does not appear in new streams, its weight will be gradually decayed to zero. Similarly, the edge weight between nodes u and v in $G_{t'}(V', E')$ is updated as

$$g_{t'}(u,v) = \begin{cases} \alpha^{-\lambda(t'_{(u,v)} - t_{(u,v)})} g_t(u,v) + \\ \qquad \sum\limits_{d \in S_{t\to t'}} I(u \in d \text{ and } v \in d) & \text{if } e_{u,v} \in S_{\leq t} \text{ and } e_{u,v} \in S_{t\to t'} \\ \alpha^{-\lambda(t'_{(u,v)} - t_{(u,v)})} g_t(u,v) & \text{if } e_{u,v} \in S_{\leq t} \text{ and } e_{u,v} \notin S_{t\to t'} \\ \sum\limits_{d \in S_{t\to t'}} I(u \in d \text{ and } v \in d) & \text{if } e_{u,v} \notin S_{\leq t} \text{ and } e_{u,v} \in S_{t\to t'}, \end{cases} \tag{5}$$

where $t'_{(u,v)}$ and $t_{(u,v)}$ denote the last appearance time of edge between nodes u and v in $S_{\leq t}$ and $S_{t\to t'}$, respectively. According to Eq. 4 and Eq. 5, the weights for nodes and edges in the word relation network will gradually decrease to zero if the corresponding nodes and edges do not appear in the newly arriving data stream $S_{t\to t'}$. Therefore, the core term groups for the outdated topics in word relation network will not be searched out because of the weak closeness and thus the outdated topics will gradually disappear. Meanwhile, if new events or topics appear in the short text stream, the word relation network will be updated and thus new core term groups can be detected.

4.5. *Real-time short text stream clustering*

4.5.1. *Predicting cluster index*

To predict cluster indexes for the streaming short texts in real time, the core term groups discovered from the word relation network are used as virtual cluster centers to absorb similar short texts. Formally, for a short text stream, $S_{\leq t} = \{\ldots, d_{t-2}, d_{t-1}, d_t\}$, and its corresponding word relation network $G_t(V, E)$, we exploit Algorithm 1 on, $G_t(V, E)$ to get a list of core term groups denoted as K_t. For each core term group $k_j \in K_t$, if it is a new core term group that will not be merged into the core term set field pd of the current text cluster structures, we initialize a new text cluster structure as CA_t^I by setting $CA_t^I.pd = k_j$, where I is a global cluster index counter and will increase when new cluster abstract appears. Assume that we have a list of text cluster structures at time t denoted as $\overrightarrow{CA_t}$, the cluster membership for short text d_i in batch $S_{\leq t}$ is predicted as follows:

$$c_i = \arg\max_{k \in \overrightarrow{CA_t}} |d_i \cap CA_t^k.pd|, \tag{6}$$

where k denotes the index of cluster abstract at time t and c_i represents the cluster index for d_i. For each short text assigned into group CA_t^k, we update its corre-

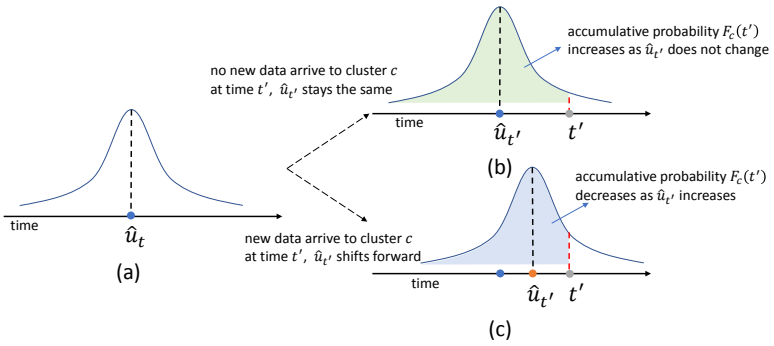

Fig. 5: Illustration of deciding the freshness of a short text cluster using the normal distribution. Fig (a) denotes that the mean of the estimated distribution is u_t at time t. Fig (b) denotes that the mean $u_{t'}$ does not change if no new data grouped into this cluster at time t'. Fig (c) denotes that the mean $u_{t'}$ changes when new data grouped into this cluster at time t'.

sponding fields such as the number of short texts and the temporal statistics. Eq. 6 indicates that short texts will be put into the cluster that has the largest number of core terms overlapping with the virtual cluster center. Eq. 6 defines an effective group forming strategies for real-time short text clustering and the time complexity denoted as $\mathcal{O}(|\overrightarrow{CA_t}|)$, which is linearly increasing with the number of *cluster abstracts* at time t. When the query window moves to time t', we update the $\overrightarrow{CA_t}$ as $\overrightarrow{CA_{t'}}$ by merging new core term groups discovered from the newly arrived data stream $S_{t\rightarrow t'}$.

4.5.2. *Merging similar core term groups*

After we update the word relation network from G_t to $G_{t'}$ for capturing changes in the stream, we need to search the core term groups in $G_{t'}$. Here, one of the issues is that the core term groups in G_t may still exist in $G_{t'}$ when the corresponding topics are active. We assume that the previous cluster structures are $\overrightarrow{CA_t}$ and the newly discovered core term groups from $G_{t'}$ are $K_{t'}$. If one core term group, $k_i \in K_{t'}$, has strong overlapping with the core term set field, pd, of one of the cluster structures in $\overrightarrow{CA_t}$, then k_i will be merged into the existing cluster structures. Otherwise, we initialize a new text cluster structure for k_i and increase the global cluster counter, I. To judge if a core term group searched from $G_{t'}$ is new, we use an efficient strategy that merges a new core term group into existing cluster structures if the new core term group is formed by the same node that has the highest node weight in the corresponding core term group (see in Fig. 4 that the terms with red color would absorb other core terms).

4.5.3. *Deleting outdated clusters*

Due to the dynamic nature of short texts, some topics may disappear after a period of time, which indicates that some text clusters would not be updated after some time. Hence, it is necessary to remove the outdated cluster abstract to improve the efficiency of Eq. 6. Here, we estimate the accumulated arrival time probability of short texts for a cluster to find out the outdated cluster structures. Specifically, we assume that the arriving time of text data for a short text cluster C, is normally distributed and we exploit the accumulative probability to determine the freshness of the cluster. Fig. 5 illustrates how to use probability density to judge the freshness of the cluster. we use two unbiased estimators to estimate the mean and deviation for the normal distribution, $\mathcal{N}_C = (\mu, \sigma^2)$. Suppose that the timestamp of the short texts in cluster C is $[t_1, t_2, \ldots, t_n]$. Then, the unbiased mean estimation for $\mathcal{N}_C$ is defined as $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} t_i$; the unbiased variance estimation for $\mathcal{N}_C$ is defined as $\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(t_i - \hat{\mu})^2$, which can also be derived as $\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n} t_i^2 - \frac{n}{n-1}\hat{\mu}^2$. Remember that in the definition for cluster abstract (CA), we extract the temporal

Algorithm 2: EWNStream+ for real time short text stream clustering.

Input: short text stream S, parameters: α, β.
Output: cluster labels for short texts in S.

```
while !S.end() do
    get the t-th batch of short texts D_t from S
    if t == 1 then
        create G_1(V, E) from D_1
    end
    else
        update G_{t-1} to G_t according to Eq. 4 and Eq. 5.
    end
    // use fast core term group searching in Algorithm 1.
    K_t = FCTGS(G_t, α, β)
    if t==1 then
        initialize CA_1 from K_1
    end
    else
        update field pd of CA_t from K_t
    end
    for d_{i,t} ∈ D_t do
        predict the cluster membership of d_{i,t} according to Eq. 6.
        update fields n, st_1 and st_2 for CA_t
    end
    delete outdated CA according to Eq. 7 and Eq. 8
end
```

statistics from the arrival timestamps of short texts in C. The temporal fields st_1 and st_2 of CA can be naturally used in the estimators of $\mathcal{N}_C$. Thus, $\hat{\mu}$ and σ^2 can be revised as

$$\begin{cases} \hat{\mu} = \frac{st_1}{n} \\ \sigma^2 = \frac{st_2}{n-1} - \frac{{st_1}^2}{n(n-1)} \end{cases} \tag{7}$$

After we have the estimated arrival timestamp probability distribution for a cluster, the problem of detecting the outdated cluster is transferred to calculate the accumulative probability:

$$F_C(t') = \int_{-\infty}^{t'} \mathcal{N}_c(x|\hat{\mu}, \hat{\sigma}^2)\ dx, \tag{8}$$

where t' denotes the end time for query window $Q = [t, t']$. For a given cluster abstract $CA_{t'}^j$, if its $F_j(t')$ is greater than this threshold, then it indicates that no new short texts are allocated into this cluster and we remove the cluster abstract CA_t^j.

The EWNStream+ is shown in Algorithm 2 for short text stream clustering. To simplify the process, we fix the query window size to process the short text stream with a batch mode. Each time, a batch of short texts is extracted from the short text stream, S. The word relation network, G, is constructed at the first batch and is updated to capture the changing of data in the stream when new batches arrive. Lines 2-8 are used to create or update the word relation network, G. We use FCTGS to discover the core term groups at line 10. Then, we initialize or update the text cluster structures based on the core term groups to cluster short texts from line 11 to line 16. The cluster memberships of short texts are predicted at lines 17-20. Then, we check the estimated probability distribution to delete the outdated clusters at line 21.

5. Experiments

In this section, we introduce our experimental setup in section 5.1 and study the optimal parameter settings for EWNStream+ in section 5.2. Then, we evaluate the accuracy, efficiency and effectiveness of EWNStream+ in sections 5.3, 5.4 and 5.5, respectively.

Table 1: Statistics of the adopted dataset.

Dataset	#tweets	#terms	avg-length	#clusters
TweetSet	143, 971	66, 288	7.44	16

5.1. *Experimental setup*

5.1.1. *Dataset*

Our experiments are conducted on a real-word short text dataset crawled from Twitter.

To evaluate the clustering quality of the proposed method, we are motivated to find the ground truth cluster label for each tweet. To do this, we exploit an online database [a] that records the major historical events happened in the world. We use the keywords provided by the database for each event as queries to retrieve the relevant tweets from the raw data provided by [50]. According to our previous work [51], many raw tweets are not related to news or events. Therefore, we remove those noisy tweets for improving both clustering accuracy and efficiency. we construct a dataset with 16 categories and totally 143, 971 tweets posted in June 2009. We name this dataset as *TweetSet* in this paper.

To clean the raw tweets, we conduct several pre-processing steps: (1) converting letters into lowercase; (2) stemming; (3) removing stop words and non-English characters; (4) removing words with length less than 3; (5) removing document with length less than 3. Table 1 shows the basic statistics of the adopted dataset.

5.1.2. *Evaluation metrics*

We adopt an external metric: Normalized Mutual Information (NMI) [8, 30] to evaluate the clustering quality. NMI is a popular metric that measures the similarity between the ground truth cluster partition of the dataset and the predicted cluster partition of the dataset. Let $\mathcal{C} = \{c_1, \cdots, c_P\}$ denotes the ground truth cluster partitions for the dataset and $\Omega = \{\omega_1, \cdots, \omega_Q\}$ denotes the predicted partitions for the dataset. Normalized Mutual Information is formally defined as follows:

$$\mathrm{NMI}(\mathcal{C}, \Omega) = \frac{\sum_{i,j} \frac{|w_i \cap c_j|}{N} \log \frac{N|w_i \cap c_j|}{|w_i||c_j|}}{(-\sum_i \frac{|w_j|}{N} \log \frac{|w_i|}{N} - \sum_j \frac{|c_j|}{N} \log \frac{|c_j|}{N})/2},$$

where N is the number of documents in the dataset. Note that, the NMI value ranges from 0 to 1 and larger NMI value indicates better clustering quality.

5.1.3. *Counterpart methods*

We compare the proposed *EWNStream+* with the following state-of-the-art methods:

- **EWNStream** [37]. This method is the preliminary version of EWNStream+. Compared with EWNStream+, EWNStream does not consider the fading of word relations and clusters in the stream.

[a] https://www.onthisday.com/

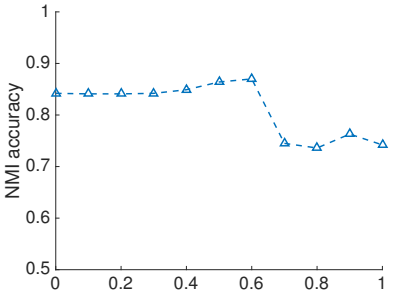

Fig. 6: Parameter analysis for α

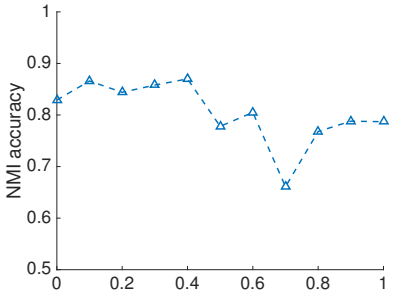

Fig. 7: Parameter analysis for β

- **MStream** [30]. This method adopts the Dirichlet Process Mixture Model (DPMM) to process stream documents. DPMM exploits the Dirichlet Process mechanism to automatically capture the change of the clusters in the streaming documents.
- **Sumblr** [7]. This method compresses the information of tweets into a dynamic statistical data structure called Tweet Cluster vector (TCV) and uses TCVs to represent different clusters. The newly coming tweets are either merged into the current clusters or start a new cluster based on the distance between tweets and TCVs.
- **Dynamic topic model (DTM)** [27]. This method is an extension of Latent Dirichlet Allocation (LDA) [35] to infer the dynamic topic distribution for the streaming corpus. DTM adopts multiple LDA models to process the documents in different time spans.

The parameters for the counterpart methods are well tuned. For MStream, we set $\alpha = 0.03$ and $\beta = 0.03$. For DTM, we set $\alpha = 0.01$. The number of iterations for MStream and DTM is set as 10. The topic number k of DTM is set as 20. For Sumblr, we set $\beta = 0.1$. The number of batches is set as 16 for all of the methods if not specified.

5.2. *Optimal parameters*

We analyze the optimal parameter settings of α and β for EWNStream+. Figs. 6 and 7 show the NMI accuracy of EWNStream+ under different α and β settings. Specifically, Fig. 6 shows the best setting for α when fixing β. α controls the weight of Jaccard similarity and co-occurrence similarity between two terms. As we can see, $\alpha = 0.6$ leads to the best NMI accuracy, which indicates a balance between the two types of similarities.

Similarly, Fig. 7 shows the influence of β on the performance of EWNStream+ by fixing α. As we can see, the higher β leads to the worse clustering performance. The best performance appears when β is around 0.4. From Figs. 6 and 7, we can make a conclusion that the best parameter settings are $\alpha = 0.6$ and $\beta = 0.4$.

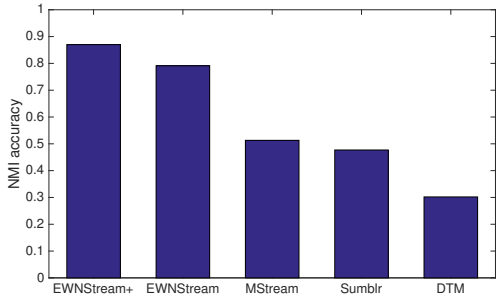

Fig. 8: Clustering performance analysis on TweetSet data.

5.3. *Accuracy evaluation*

In Fig. 8, we compare the NMI accuracy of different methods on the TweetSet dataset with the same batch number setting. Here, the batch number for EWNStream+, EWNStream, MStream and DTM is set as 16.

From Fig.8, the overall trend is that the proposed EWNStream+ and EWNStream are significantly better than the other three methods (MStream, Sumblr and DTM) in TweetSet. Take EWNStream as an example, its NMI accuracy is around 0.8, which is 30% higher than MStream. Meanwhile, EWNStream+ is around 7% better than EWNStream, which shows the improvement when taking the fading policy of the word relations and topics into consideration. MStream ranks as the third among all methods for TweetSet, which is around 0.5. MStream adopts a Dirichlet Process mechanism to automatically adjust cluster number for the short text streams, but the Dirichlet Process relies on a rich word co-occurrence patterns to generate a new cluster. However, most word pairs in short text corpus show quite sparse word co-occurrence relations. The NMI accuracies of Sumblr and DTM are the worst, around 0.47 and 0.30, respectively. Sumblr adopts the vector space model to represent short texts for the similarity calculation, which is not accurate. DTM also relies on a rich word co-occurrence pattern to accurately infer the topic distribution, its performance is influenced by the sparsity of short texts.

Fig. 9 shows the influence of batch number setting for the accuracy of different methods. The batch number setting varies from 2 to 16. As we can see, the performance of the proposed method is stable with different batch numbers and achieves around 0.85 NMI accuracy and outperforms its counterpart at different batch number settings. DTM shows the worse performance among all the methods with around 0.35 NMI accuracy at all different batch settings. MStream shows slightly better performance than Sumblr with around 0.50 NMI accuracy.

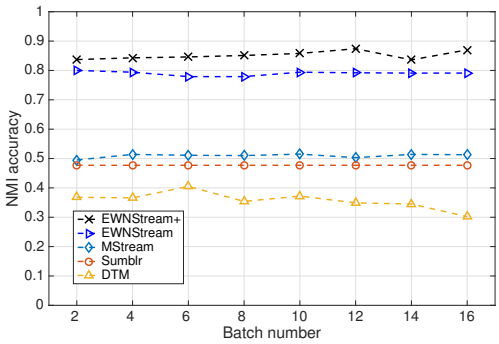

Fig. 9: The influence of batch number setting on clustering accuracy.

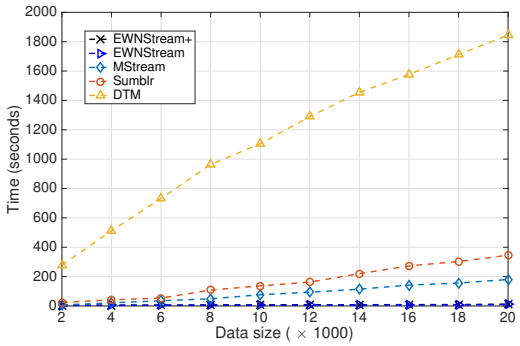

Fig. 10: The influence of dataset size on clustering efficiency.

5.4. *Efficiency evaluation*

Fig. 10 shows the efficiency comparison among all methods. All of the five methods are implemented in Python and tested on the same computing server with 1.5GHz CPU and 64 GB memory. The execution time is calculated with cleaned datasets and focuses on the short text stream clustering methods, and thus excludes the pre-processing steps, such as term stemming and stop word stemming. All of the methods are tested on the same datasets with incrementally increased data size for fair comparison.

We can see that the execution time of the five methods is linearly increasing with the size of the dataset. The overall trend is that the EWNStream+ and EWNStream are apparently faster than the three counterpart methods (MStream, Sumblr and DTM). Meanwhile, EWNStream+ has the similar time costs as EWNStream; this indicates that incorporating the outdated cluster abstract detection policy does not lead to the increasing of time cost. Sumblr shows worse time efficiency than

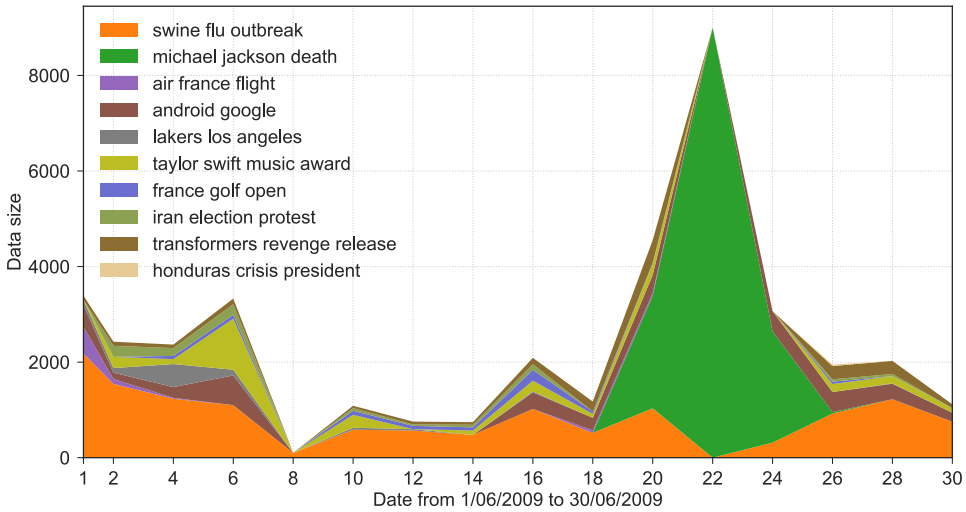

Fig. 11: The evolution of different short text clusters.

MStream due to the average cosine similarity computing between the new tweet and different candidate clusters. Among the four methods, DTM has the worst time efficiency. Both DTM and MStream need to process a text document multiple times. But DTM samples a topic at the word level in each document and MStream samples a topic at the document level. Hence, MStream is more efficient than DTM. EWNStream+ can be maximum of 30 times faster than DTM.

Table 2: The core terms evolution for two example topics. The terms with underline are identified as new appeared core terms.

Topic: swine flu outbreak	Topic: michael jackson death
[**06.01-06.05**] flu, swine, pandemic, virus, health, organization, vaccine, declared	-
[**06.07-06.10**] flu, swine, pandemic, virus, health, school, level, patient	-
[**06.11-06.15**] flu, swine, pandemic, ship, cruise, cases, death	[**06.18-06.22**] michael,jackson,died, holy,heard,life,confirm
[**06.16-06.18**] flu, swine, pandemic, pig, summer,philippines	[**06.22-06.26**] michael,jackson,died, god, rest, peace, hospital
[**06.24-06.30**] flu, swine, pandemic, million,hit	-

5.5. *Effectiveness analysis*

Fig. 11 shows the evolution of various short text clusters from the date 1/06/2009 to the date of 30/06/2009 in regards to cluster size. The extracted clusters cover various topics such as political events, sports, celebrities. As we can see in Fig 11 that different topics have various evolution patterns. Take the topic of flu that supported by core terms (‘‘swine’’,‘‘flu’’) as an example, it lasts a long period of time in the stream and has been hotly discussed at different times. This may because that the flu may periodically occur and cause panic to people (shown as the orange area in Fig. 11). Another obvious topic in the stream is the death of a famous singer (‘‘michael’’,‘‘jackson’’). From 18/06 to 26/06, lots of tweets were posted to discuss this event (shown as the green area in Fig. 11). Table 2 represents the core term evolution of the above mentioned two clusters. We can see that the two topics are supported by several core terms. As time lapses, the topic may gradually evolve as different aspects of the topic. For example, in the topic about the death of Michael Jackson, users were surprised and could not believe the sudden loss of the famous singer. Hence, during the time period from 18/06 to 22/06, people used the term like ‘‘confirm’’ and ‘‘holy’’ to express their surprise. When the news has been confirmed, the users tend to use terms like ‘‘rest’’ and ‘‘peace’’ to express their emotions.

6. Discussion

The efficiency and accuracy are two important aspects to consider in short text stream clustering due to the huge data size, sparsity and topic drift problems. From the experimental results, we can see that the proposed EWNStream+ and EWNStream outperform the other three methods significantly in terms of both accuracy and efficiency. The results indicate that when considering the processing for short text corpora, it is necessary to adopt new methodologies for overcoming the apparently sparse and noise attributes of short text data. The proposed methods avoid the representation of short text corpus and only focus on the term relations and term statistics that lead to the differences between the general terms and core terms. The benefits are that the proposed methods not only can avoid the distance measurement between short texts and the cluster centers but also lead to naturally discover of the topic terms to represent the events in the text streams. Furthermore, the efficiency of the proposed methods also show that the evolutionary and fast core term searching strategies on the word relation network are useful to reach the nearly real-time clustering for the short text streams.

One of the aspects that may need more consideration in the proposed methods is the word relation network scalability issue. As we know that the dictionary size for social media corpora is huge and always increasing due to the facts that lots of new terms or buzzwords would appear in social media. Hence, the word relation network in the proposed method may become quit large in a specific time period.

This may influence the efficiency of the core term searching and thus increase the whole processing time. Recently, the development of graph neural network [52] shows great potential to mine the structural patterns and could be applied to lots of applications such as community detection and node classification. Therefore, one of the promising aspects to further improve the proposed methods could be adopting the graph neural network into the core term group discovery for clustering short texts.

7. Conclusion

In this paper, we have proposed EWNStream+, which exploits an evolutionary word relation network to handle the challenges of clustering short texts in streams. The bi-weighted word relation network was constructed based on the aggregated word relations from batches of documents in the short text streams. To capture the content drift in short text streams, we dynamically update the word relation network with the newly arrived data. Meanwhile, we also have introduced a decay strategy to naturally filter the old word relation patterns to remove the core term groups that may not be supported by new data. To cluster short texts, we have developed a fast core term searching strategy to extract the representative terms for different topics from the word relation network and used the core terms as the cluster centers to absorb semantically similar short texts into clusters. In this way, we can avoid difficulties in short text representation and distance computing. Extensive experiments on a real-world short text stream dataset show that our approach can achieve better clustering accuracy and better time efficiency for short text streams than several counterpart methods.

8. Acknowledgments

This work was partially supported by Australia Research Council (ARC) Grants (No. DE140100387 and No. DP190100587).

References

1. X. Fu, J. Lee, C. Yan, and L. Gao, "Mining newsworthy events in the traffic accident domain from chinese microblog," *International Journal of Information Technology & Decision Making*, vol. 18, no. 02, pp. 717–742, 2019.
2. Y. Wang, X. Zhang, and Z. Wang, "A proactive decision support system for online event streams," *International Journal of Information Technology & Decision Making*, vol. 17, no. 06, pp. 1891–1913, 2018.
3. N. Wang, S. Ke, Y. Chen, T. Yan, A. Lim, *et al.*, "Textual sentiment of chinese microblog toward the stock market," *International Journal of Information Technology & Decision Making*, vol. 18, no. 02, pp. 649–671, 2019.
4. Y. Shi, L. Zhu, W. Li, K. Guo, Y. Zheng, *et al.*, "Survey on classic and latest textual sentiment analysis articles and techniques," *International Journal of Information Technology & Decision Making*, vol. 18, no. 04, pp. 1243–1287, 2019.

5. H. Yin, S. Yang, and J. Li, "Detecting topic and sentiment dynamics due to covid-19 pandemic using social media," *arXiv preprint arXiv:2007.02304*, 2020.
6. J. Zhou, S. Yang, C. Xiao, and F. Chen, "Examination of community sentiment dynamics due to covid-19 pandemic: a case study from australia," *arXiv preprint arXiv:2006.12185*, 2020.
7. L. Shou, Z. Wang, K. Chen, and G. Chen, "Sumblr: continuous summarization of evolving tweet streams," in *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pp. 533–542, 2013.
8. S. Yang, G. Huang, and B. Cai, "Discovering topic representative terms for short text clustering," *IEEE Access*, vol. 7, pp. 92037–92047, 2019.
9. M. Garcia Alvarez, J. Morales, and M.-J. Kraak, "Integration and exploitation of sensor data in smart cities through event-driven applications," *Sensors*, vol. 19, no. 6, p. 1372, 2019.
10. W. Feng, C. Zhang, W. Zhang, J. Han, J. Wang, C. Aggarwal, and J. Huang, "Streamcube: hierarchical spatio-temporal hashtag clustering for event exploration over the twitter stream," in *2015 IEEE 31st International Conference on Data Engineering*, pp. 1561–1572, IEEE, 2015.
11. Y. Zhao, S. Liang, Z. Ren, J. Ma, E. Yilmaz, and M. de Rijke, "Explainable user clustering in short text streams," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 155–164, ACM, 2016.
12. T. Sakaki, M. Okazaki, and Y. Matsuo, "Tweet analysis for real-time event detection and earthquake reporting system development," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 4, pp. 919–931, 2012.
13. Y. Takeichi, K. Sasahara, R. Suzuki, and T. Arita, "Twitter as social sensor: Dynamics and structure in major sporting events," in *Artificial Life Conference Proceedings 14*, pp. 778–784, MIT Press, 2014.
14. G. Kou, X. Chao, Y. Peng, F. E. Alsaadi, and E. Herrera-Viedma, "Machine learning methods for systemic risk analysis in financial sectors," *Technological and Economic Development of Economy*, vol. 25, no. 5, pp. 716–742, 2019.
15. H. Wang, G. Kou, and Y. Peng, "Multi-class misclassification cost matrix for credit ratings in peer-to-peer lending," *Journal of the Operational Research Society*, pp. 1–12, 2020.
16. G. Kou, Y. Peng, and G. Wang, "Evaluation of clustering algorithms for financial risk analysis using mcdm methods," *Information Sciences*, vol. 275, pp. 1–12, 2014.
17. F. Shen, X. Zhao, and G. Kou, "Three-stage reject inference learning framework for credit scoring using unsupervised transfer learning and three-way decision theory," *Decision Support Systems*, vol. 137, p. 113366, 2020.
18. Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: concepts, methodologies, and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 5, no. 3, p. 38, 2014.
19. J. Jiang, S. Wen, S. Yu, Y. Xiang, and W. Zhou, "Rumor source identification in social networks with time-varying topology," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 1, pp. 166–179, 2016.
20. T. Li, G. Kou, Y. Peng, and Y. Shi, "Classifying with adaptive hyper-spheres: An incremental classifier based on competitive learning," *IEEE transactions on systems, man, and cybernetics: systems*, 2017.
21. J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Transactions on Knowledge and Data Engineering*, 2018.
22. C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "A framework for clustering evolving

data streams," in *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pp. 81–92, VLDB Endowment, 2003.
23. F. Cao, M. Estert, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in *Proceedings of the 2006 SIAM international conference on data mining*, pp. 328–339, SIAM, 2006.
24. S. Zhong, "Efficient streaming text clustering," *Neural Networks*, vol. 18, no. 5-6, pp. 790–798, 2005.
25. M. R. H. Rakib, N. Zeh, and E. Milios, "Short text stream clustering via frequent word pairs and reassignment of outliers to clusters," in *Proceedings of the ACM Symposium on Document Engineering 2020*, pp. 1–4, 2020.
26. N. C. Crossman, S. M. Chung, and V. A. Schmidt, "Stream clustering and visualization of geotagged text data for crisis management," in *2019 International Conference on Data and Software Engineering*, pp. 1–6, IEEE, 2019.
27. D. M. Blei and J. D. Lafferty, "Dynamic topic models," in *Proceedings of the 23rd international conference on Machine learning*, pp. 113–120, ACM, 2006.
28. H. Amoualian, M. Clausel, E. Gaussier, and M.-R. Amini, "Streaming-lda: A copula-based approach to modeling topic dependencies in document streams," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 695–704, ACM, 2016.
29. S. Liang, E. Yilmaz, and E. Kanoulas, "Dynamic clustering of streaming short documents," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 995–1004, ACM, 2016.
30. J. Yin, D. Chao, Z. Liu, W. Zhang, X. Yu, and J. Wang, "Model-based clustering of short text streams," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (New York, NY, USA), pp. 2634–2642, ACM, 2018.
31. X. Zhou and L. Chen, "Event detection over twitter social media streams," *The VLDB journal*, vol. 23, no. 3, pp. 381–400, 2014.
32. J. Chen, Z. Gong, and W. Liu, "A nonparametric model for online topic discovery with word embeddings," *Information Sciences*, vol. 504, pp. 32–47, 2019.
33. Z. Lu, H. Tan, and W. Li, "An evolutionary context-aware sequential model for topic evolution of text stream," *Information Sciences*, vol. 473, pp. 166–177, 2019.
34. K. Liu, A. Bellet, and F. Sha, "Similarity learning for high-dimensional sparse data," in *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics, 2015, San Diego, CA, USA.*, 2015.
35. D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *The Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.
36. J. Jiang, S. Wen, S. Yu, Y. Xiang, W. Zhou, and H. Hassan, "The structure of communities in scale-free networks," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 14, p. e4040, 2017.
37. S. Yang, G. Huang, X. Zhou, and Y. Xiang, "Dynamic clustering of stream short documents using evolutionary word relation network," in *International Conference on Data Science*, pp. 418–428, Springer, 2019.
38. G. Huang, J. He, Y. Zhang, W. Zhou, H. Liu, P. Zhang, Z. Ding, Y. You, and J. Cao, "Mining streams of short text for analysis of world-wide event evolutions," *World wide web*, vol. 18, no. 5, pp. 1201–1217, 2015.
39. J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. De Carvalho, and J. Gama, "Data stream clustering: A survey," *ACM Computing Surveys*, vol. 46, no. 1, p. 13, 2013.
40. P. Kranen, I. Assent, C. Baldauf, and T. Seidl, "The clustree: indexing micro-clusters

for anytime stream mining," *Knowledge and information systems*, vol. 29, no. 2, pp. 249–272, 2011.
41. M. Carnein, D. Assenmacher, and H. Trautmann, "Stream clustering of chat messages with applications to twitch streams," in *International Conference on Conceptual Modeling*, pp. 79–88, Springer, 2017.
42. T. Iwata, S. Watanabe, T. Yamada, and N. Ueda, "Topic tracking model for analyzing consumer purchase behavior," in *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.
43. L. M. Q. Abualigah, *Feature selection and enhanced krill herd algorithm for text document clustering*. Springer, 2019.
44. L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "Hybrid clustering analysis using improved krill herd algorithm," *Applied Intelligence*, vol. 48, no. 11, pp. 4047–4071, 2018.
45. L. M. Q. Abualigah and E. S. Hanandeh, "Applying genetic algorithms to information retrieval using vector space model," *International Journal of Computer Science, Engineering and Applications*, vol. 5, no. 1, p. 19, 2015.
46. L. M. Abualigah and A. T. Khader, "Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering," *The Journal of Supercomputing*, vol. 73, no. 11, pp. 4773–4795, 2017.
47. L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "A new feature selection method to improve the document clustering using particle swarm optimization algorithm," *Journal of Computational Science*, vol. 25, pp. 456–466, 2018.
48. L. M. Abualigah, A. T. Khader, E. S. Hanandeh, and A. H. Gandomi, "A novel hybridization strategy for krill herd algorithm applied to clustering techniques," *Applied Soft Computing*, vol. 60, pp. 423–435, 2017.
49. G. Kou, P. Yang, Y. Peng, F. Xiao, Y. Chen, and F. E. Alsaadi, "Evaluation of feature selection methods for text classification with small datasets using multiple criteria decision-making methods," *Applied Soft Computing*, vol. 86, p. 105836, 2020.
50. J. Yang and J. Leskovec, "Patterns of temporal variation in online media," in *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, (New York, NY, USA), pp. 177–186, ACM, 2011.
51. Y. You, G. Huang, J. Cao, E. Chen, J. He, Y. Zhang, and L. Hu, "Geam: A general and event-related aspects model for twitter event detection," in *International Conference on Web Information Systems Engineering*, pp. 319–332, Springer, 2013.
52. Y. Ma, Z. Guo, Z. Ren, J. Tang, and D. Yin, "Streaming graph neural networks," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 719–728, 2020.