Received June 9, 2019, accepted July 3, 2019, date of publication July 8, 2019, date of current version July 25, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2927345

Discovering Topic Representative Terms for Short Text Clustering

SHUIQIAO YANG, GUANGYAN HUANG, AND BORUI CAI
School of Information Technology, Deakin University, Melbourne, VIC 3125, Australia

Corresponding author: Guangyan Huang (guangyan.huang@deakin.edu.au)

This work was supported in part by the Australian Research Council (ARC) under Grant DE140100387.

ABSTRACT Clustering short texts are one of the most important text analysis methods to help extract knowledge from online social media platforms, such as Twitter, Facebook, and Weibo. However, the instant features (such as abbreviation and informal expression) and the limited length of short texts challenge the clustering task. Fortunately, short texts about the same topic often share some common terms (or term stems), which can effectively represent a topic (i.e., supported by a cluster of short texts), and we also call them topic representative terms. Taking advantage of topic representative terms, it is much easier to cluster short texts by grouping short texts into the most similar topic representative term groups. This paper provides a novel topic representative term discovery (TRTD) method for short text clustering. In our TRTD method, we discover groups of closely bound up topic representative terms by exploiting the closeness and significance of terms. The closeness of the topic representative terms is measured by their interdependent co-occurrence, and the significance is measured by their global term occurrences throughout the whole short text corpus. The experimental results on real-world datasets demonstrate that TRTD achieves better accuracy and efficiency in short text clustering than the state-of-the-art methods.

INDEX TERMS Short text, clustering, topic representative terms.

I. INTRODUCTION

Short text documents are increasingly available with the advancement of online social media platforms, such as Twitter, Facebook and Weibo, etc. Clustering short text documents is one of the most significant text analysis methods to help extract knowledge from the abundant text data on the internet, such as news titles and tweets. The applications include event discovery [1], social spam detection [2], sentimental analysis [3], etc. However, according to many researchers [4]–[6], short text clustering is more challenging than the regular text clustering. It is due to the instant features (e.g., abbreviation and informal expression) and shortness of the text that brings sparsity, noise and high dimensionalities in the process of text analytics. Table 1 shows three examples of short text documents. As we can see, short texts contain lots of noise and provide limited contextual clues for applying traditional data mining techniques. Therefore, many adapted approaches were proposed for short text clustering in recent years.

TABLE 1. Examples of three short texts from Twitter. The length of those documents is limited and each text contains informal terms and abbreviations; these challenge clustering short text.

Tweet 1: iPhone 3GS Processor / RAM Specs Revealed? If These Are Right - Hurrah!
Tweet 2: Ultimate Festival Band? We're at Number 3!Keep em' comin!
Tweet 3: Volkswagen New Beetle 1.8T 193 PK Highline SUPERTUNING!

Existing short text clustering methods broadly fall into two categories: representation-based methods [4], [7]–[9] and model-based methods [5], [6], [10]. The representation-based methods focus on using enriched or compact features to represent short text documents to overcome the sparse issues of using vector space model for short text representation. Then, the conventional clustering methods, such as K-Means [11], are adopted to group short texts. Example methods in this category include: Wiki_Method [7], DLDA [8], STC^2 [4], LSI [12], etc. These methods try to solve the sparsity of raw word feature representation through enriching short text features or extracting more compact latent features to represent short texts. For example, Wiki_Method enriches short text representation with additional features from Wikipedia; DLDA transfers topic relevant knowledge from auxiliary long texts to short texts for topic distribution vector learning; STC^2 adopts pre-trained word embeddings [13] and Convolutional neural network [14] to learn deep feature representation for short

The associate editor coordinating the review of this manuscript and approving it for publication was Laurence T. Yang.

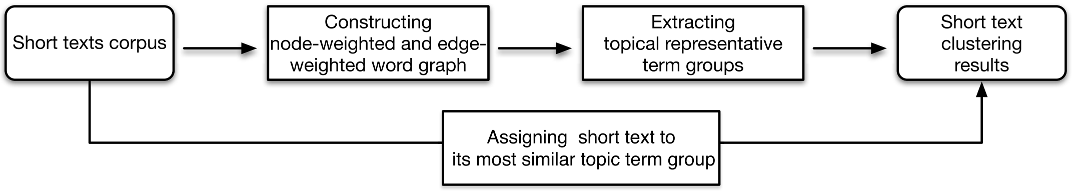

Short texts corpus
Constructing node-weighted and edge-weighted word graph
Extracting topical representative term groups
Short text clustering results
Assigning short text to its most similar topic term group

FIGURE 1. The architecture of TRTD for short text clustering.

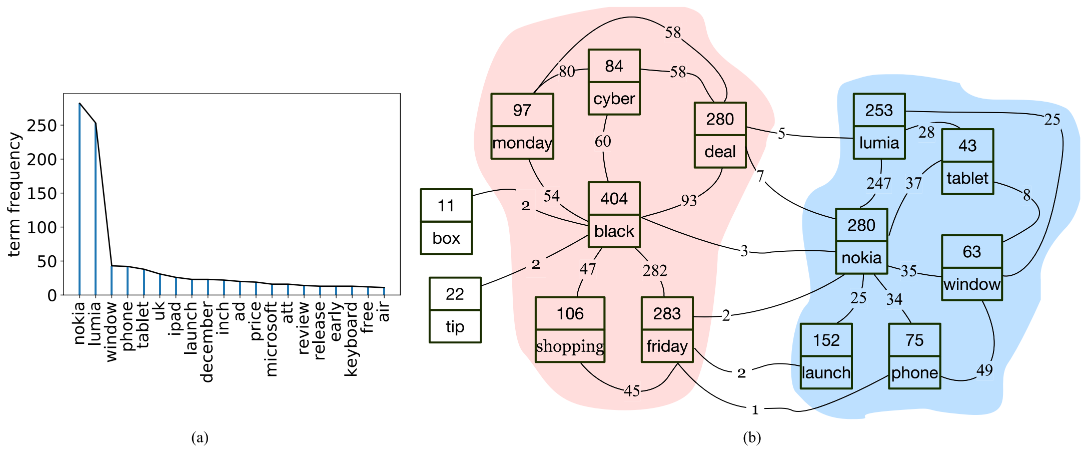


FIGURE 2. (a) Term frequency distribution in the cluster of short texts on topic "nokia lumia". The term frequency distribution shows a long-tail phenomenon, where several terms have significant higher frequency. The short texts share two common terms: "*nokia*" and "*lumia*". (b) Illustration of a node-weighted and edge-weighted word graph. Each node represents a term with corpus-level term frequency as node weight. Each edge represents the co-occurrence relation between two terms with corpus-level co-occurrence frequency as edge weight. The topic representative terms within each cluster are closely bound up in regards to the nodes' weight and edges' weight.

texts. However, learning accurate representation for short texts is not easy since the short text is noisy, sparse and lack of context. What is more, the methods adopt external knowledge base may be inflexible when the relevant contextual contents from external resources are rare.

The model-based methods are designed with new clustering strategies for short text documents to avoid the sparseness issues in short text representation. Example methods include: GSDMM [6], GPU-DMM [15], BTM [9], TermCut [10], WordCom [5], etc. For instance, Ni *et al.* [10] proposed a bisecting clustering method, TermCut, to extract one core term for each short text cluster. However, similar to TextRank [16], a single representative term is often insufficient to determine the topic of a short text cluster. Jia *et al.* [5] proposed a method, WordCom. It first separates words into communities by using a K-Means based community detection method, k-rank-D [17]. One community represents one topic, and all the words in a community are treated as the representative terms of a topic. Then, the word communities will be used to infer cluster membership of short texts. However, the word community often involves noise terms (i.e., low-frequency terms).

Inspired by the previous studies [5], [10], [18], which use words relation network to address the difficulties in short text clustering, in this paper, we propose a novel topic representative terms discovery (TRTD) method to find those significant terms that are closely bound up with each other as a group of topic representative terms for short text clustering. As we discussed before, shortness is a critical challenge for short text clustering. But every coin has two sides, within a limited length, short texts have to be very concise: using few but highly concentrated topic representative terms to express the main idea of the underlying topic. In fact, we have observed that any cluster of short texts about the same topic often share some common terms (such as "nokia" and "lumia" in Fig. 2 (a)), and the topic representative terms within each cluster are closely bound up with each other (we define the closeness of different terms using the node/edge weights in word graph as shown in Fig. 2 (b)). The proposed TRTD is based on these two key insights.

To extract topic representative term groups, we first construct a node-weighted and edge-weighted word graph (NEWG) for the corpus. Each node denotes a term, weighted by its term frequency at corpus-level, i.e., term frequencies are measured by their global term occurrences throughout the whole short text corpus. Each edge is weighted by the co-occurrence of two corresponding terms at corpus-level (see Fig. 2 (b)). NEWG aggregates the words statistics from the whole short text corpus and thus relieves the sparse context and word co-occurrence patterns of short texts at document-level. Then, we locate *seed terms* whose node weight are relatively higher than most of their neighbors. For each seed term, we extract the *closely bounded neighbor terms* that satisfying the *closeness* measurement. Each seed term and its closely bounded neighbor terms form a topic representative term group for a short text cluster. Finally, short texts are grouped into a cluster by joining the most similar topic representative term group. Fig 1 shows the architecture of TRTD for short text clustering. There are three steps in TRTD: (1) constructing node/edge-weighted word graph, (2) extracting topical representative groups and (3) clustering short texts. In step two, we extract the topic representative term groups based on the closely bound up words relation pattern.

We summarize the main contributions of this paper as follows:

- We propose a novel topic representative term discovery (TRTD) method for short text clustering. TRTD defines the closeness between terms using a node-weighted and edge-weighted word graph. TRTD overcomes shortness and sparsity challenges of short texts using the aggregated word relation network built from the whole short text corpus.
- The proposed TRTD method addresses both insufficient and noise issues in existing methods of extracting topic representative terms. TRTD can effectively find those significant terms that are closely bounded up with each other as the group of topic representative terms.
- We conduct extensive experiments on real-world datasets to demonstrate the accuracy and efficiency of the proposed TRTD for short text clustering.

The rest of this paper is organized as follows. Section II presents related work. Section III details the proposed approach. Experimental results are reported in Section IV. We finally conclude this paper in Section V.

II. RELATED WORK

In this section, we review the existing methods for short text clustering and classify these methods into two categories: representation-based methods and model-based methods.

A. REPRESENTATION-BASED METHODS

This type of methods focuses on learning effective representation vectors for short texts and exploiting traditional clustering methods such as K-Means [11] for clustering. The classic way to represent text data is via Vector Space Model (VSM) [19], where texts are represented with term weight vectors. The weight for a term is based on the term frequency in text documents. However, as short text documents are sparse and most terms only occur once in a short text. Therefore, using the Vector Space Model to represent short texts will lead to high-dimensional and sparse vectors, which is less discriminative when calculating Euclidean distances or Cosine similarities [20].

To solve the sparsity issue of representing short texts, researchers have enriched the context for short text representation. For example, Banerjee *et al.* [7] have augmented the TF-IDF representation for short text documents using relevant Wikipedia concepts. CLUTO[1] was used as the clustering engine to cluster short text documents based on the augmented TF-IDF representation. Tang *et al.* [21] have proposed an integration framework which can incorporate different language knowledge and adopted matrix factorization techniques to reduce the high dimensional representation of tweets into more compact representations. Zheng *et al.* [22] have enriched short text representation to improve short text clustering performance. In their method, short text documents are mapped from an original feature space to a hidden semantic space by add virtual term frequencies of new words in a short text document. What is more, Huang *et al.* [23] have used the concept graph for keywords expansion on short text documents. In their method, they extract keywords from the concept graph to expand short text to address the insufficiency of the keywords in short text clustering.

Recently, some artificial neural network based methods were developed for short text representation learning. Kozlowski and Rybinski [24] have used neural network based distributional semantic model for enriching the semantic meaning of short text for clustering. Similarly, Xu *et al.* [4] have proposed STC^2, which adopts deep learning techniques for short text representation learning. The core technique of STC^2 is a Convolutional Neural Network (CNN) [14], which can learn new features from short texts. As CNN is a supervised model which needs extra labels to guide the training process, Xu *et al.* transfer the term frequency vectors of short texts into binary code vectors and use them as ground-truth labels. Then, they use word embeddings [13] to represent the short text and fed it into CNN. The output of CNN are used to fit the binary codes in the training process. After CNN has been successful trained, the last hidden layer of CNN is used to learn new features for short texts and K-Means is exploited to do the clustering task.

Other methods extract the latent semantic representation for text documents. One of the classic methods is Latent Semantic Index (LSI) [12], which uses a term-document matrix to describe the occurrence patterns for a term in documents. Then, singular value decomposition is adopted to extract the latent semantic features for each documents by factorizing the term document matrix. Another famous method is Latent Dirichlet Allocation (LDA) [25], which assumes text documents are generated by a set of latent topics. LDA infers the latent topic distribution to represent the text data. Jin *et al.* [8] have proposed a Dual Latent Dirichlet Allocation (DLDA) model to transfer the topical knowledge from auxiliary long text documents to short text documents to relief the sparsity issues in short text representation. Based on the enriched topic representation, they exploit K-Means to cluster short texts. However, the challenge for DLDA is that it is hard to choose a suitable auxiliary long text corpus to enrich the topic representation of the short texts.

B. MODEL-BASED METHODS

Model-based methods focus on designing new clustering strategies for short texts and do not need to represent short texts with feature vectors that required by traditional clustering methods.

[1]http://glaros.dtc.umn.edu/gkhome/views/cluto/

For example, Yin and Wang [6] have proposed GSDMM for short text clustering by adopting the Dirichlet Multinomial Mixture (DMM) model. GSDMM is a probabilistic generative model for short text corpus. It relieves the sparse issue of short texts in clustering task with the assumption that each short text is generated by a single latent topic. The topics of short texts are inferred through Collapsed Gibbs sampling methods and used as cluster labels.

Li *et al.* [15] have proposed GPU-DMM which exploits word embedding techniques [13] to relieve the sparsity issues of short text in topic inference. Compared with GSDMM, GPU-DMM inferences the topic index for short texts by promoting the semantic similar words with the similarity information from the pre-learned word embeddings. BTM [9] infers the latent topics for short texts by explicitly modeling the generation of bi-terms in the whole short text corpus. In the aspect of reliving the sparse issue for short text in topic mining, BTM transfers the whole corpus into bag of word pairs and infers the latent topic distributions with the aggregated patterns of word co-occurrence.

Qiang *et al.* [26] have proposed a Pitman-Yor process mixture model (PYPM) based on collapsed Gibbs sampling for short text clustering. PYPM improves GSDMM by automatically determining the cluster numbers for short text dataset with Pitman-Yor process. Specifically, in the short text clustering process, a short text chooses an existing active cluster or a new cluster with the probabilities derived from the Pitman-Yor Process Mixture model.

Researchers also have proposed models to exploit word graph or text graph for short text clustering. For example, Ni *et al.* [10] have proposed TermCut, which finds one core term for one cluster by using a bisecting clustering method. TermCut models the short text dataset as a connected graph in which each node represents a short text. At each step of bisection, one core term is extracted through optimizing the clustering criterion RMcut. The RMcut value of a word is used to determine if the word shows the best cluster quality by dividing the short texts into two groups—contain or not contain the word.

Jia *et al.* have proposed WordCom [5] which adopts k-rank-D [17] on word co-occurrence network to separate words into communities which represents different topics. The discovered word communities are used to infer cluster membership for short texts. Short texts can join into a word community if the short texts have a minimum cosine distance to the word community. What is more, Jinarat *et al.* [18] have used word semantic graph for short text clustering. In their method, the word semantic graph is constructed using the semantic similarity from word embedding techniques. Similar to Termcut, short texts are clustered if they contain at least one semantic word in the same semantic subgraph.

The proposed TRTD belongs to model-based method for short text clustering since it is a new clustering strategy, which needs not to learn the representation for short texts. Compared to existing word graph based methods, TRTD has successfully addressed the problems of sparsity, insufficiency and noise issues in short text clustering by exploring the closely bound relationship in terms to extract only the topic representative term groups.

III. THE PROPOSED APPROACH

In this section, we firstly introduce basic concepts, define the short text clustering problem, and then present our proposed **TRTD**. The TRTD includes three main steps: (1) constructing a node-weighted and edge-weighted word graph (section III.B), (2) extracting topic representative term groups (section III.C), and (3) clustering short texts (section III.D).

A. BASIC CONCEPTS AND PROBLEM DEFINITION

We adopt the definitions of document and corpus in [25]:

- A document is a sequence of n terms denoted as $d = \{w_1, w_2, \cdots, w_n\}$, where w_i is the i-th term in d. As for short text documents, n is quite small compared with regular length documents.
- A corpus is a collection of m document denoted as $D = \{d_1, d_2, \cdots, d_m\}$.

We have the following definitions used in this paper:

Definition 1 (Node-Weighted and Edge-Weighted Word Graph (NEWG)): In NEWG, each node represents a term in the short text corpus with the term frequency as the node weight. Each edge of NEWG represents the co-occurrence relations of two terms in the short text corpus with the co-occurrence frequency as the edge weight. NEWG contains the structure of topical representative term groups for each short text cluster in corpus D.

Definition 2 (Seed Term): A term in NEWG is a seed term if it shows the following two attributes: (1) significantly higher node weight than most of its neighboring terms; (2) densely connected with its neighboring term. Examples of seed terms are shown in Fig. 2 (b), where nodes "black" and "nokia" are two seed terms.

Definition 3 (Affiliated Term): A term in NEWG is an affiliated term if it shows strong closeness relationships with a seed term. The relationship between two terms is denoted by edge weight and node weight.

Given a short text corpus D with m unlabeled short texts, the problem of short text clustering is to partition D into k different groups, in which short texts in the same group are more similar to each other than those in other groups.

B. NODE-WEIGHTED AND EDGE-WEIGHTED WORD GRAPH CONSTRUCTION

Given a short text corpus, D, it contains a collection of short text documents denoted as $D = \{d_1, d_2, \cdots, d_m\}$, where m is the number of documents in D. WordCom [5] also constructs a word graph for short text clustering, which extracts word communities based on word co-occurrences. But WordCom only considers edge-weighted graphs using word co-occurrences, while the significance of the words themselves is neglected. Our goal is to find groups of closely bounded significant terms using both the closeness of words in co-occurrences and the significance of individual words.

Here, we construct a node-weighted and edge-weighted graph, $G = (V, E)$, for words in the texts in D (lines 1–10 in **Algorithm 1**). The node set V consists of the distinct *terms* in D, and each node, say w, is weighted by the corresponding *corpus-level term frequency*, $f(w)$. For an arbitrary edge $e(\in E)$ linking two nodes (say, w and v), it represents the co-occurrence of the two corresponding terms, and it is weighted by the *corpus-level co-occurrence frequency* $f(w, v)$ of the two terms (see Fig. 2 (b) for an illustration). Due to the sparsity of short texts, many words co-occur only once or twice across all documents. To diminish the impact from these unusual co-occurrences, we filter them out by the following rule:

$$Rule\ 1: e(w, v) \text{ is a valid edge in graph } G, \text{ if } f(w, v) \geq \gamma.$$

C. TOPIC REPRESENTATIVE TERM GROUP DISCOVERY

In this section, we extract topic representative term groups based on graph G constructed in section III-B. Before we present the detailed algorithm, we analyze the connection of intra-cluster terms and the connection of inter-cluster terms. Intuitively, the frequent terms within a cluster are densely connected, while the terms of different clusters are loosely connected or even disconnected. Fig. 2 (b) shows the top 6 frequent terms from topics "`black friday`" and "`nokia lumia`" from a part of the *Title* dataset [6] used in this paper. As we can see, the term `black` is *the most frequent* term in the cluster "`black friday`", which acts like a **seed term** densely connected (*i.e.*, high co-occurrence) by the other 5 frequent terms (acting like `black`'s **affiliated terms**). Similar phenomenon can be observed on term `nokia` in the cluster "`nokia lumia`". Meanwhile, the frequent terms `nokia`, `lumia`, and `launch` are also connected to term `black`, but with very few *connections/supports* (*e.g.*, 3 out of 280 from `nokia`). Our proposed TRTD is to extract these seed terms and their closely bounded affiliated terms.

Formally, a **seed term** (see Definition 2) in graph G is a node, whose weight is relatively higher than most of its neighbors. The following condition gives the criteria for seed term selection:

$$Rule\ 2: w \text{ is a seed term, if } f(w) \geq f(v),\ \forall\, v \in N(w) \text{ and } v \notin SeedSet,$$

where $N(w)$ is the set of w's neighbors in G, and $SeedSet$ is a set to record the already discovered seed terms. We define a seed term's **affiliated terms** (see Definition 3) as those nodes that not only have relatively high frequency but also present sufficient supports to the seed term in graph G. The following gives the selection criteria of a seed term's affiliated terms:

$$Rule\ 3: v \text{ is an affiliated term of a seed term } w, \text{ if } \frac{f(v)}{f(w)} \geq \delta \text{ and } \frac{f(w, v)}{f(v)} \geq \theta,$$

Algorithm 1: Topic Representative Term Group Discovery

```
Input: A corpus $D$ of short text documents, parameters: $\gamma, \delta, \theta$.
Output: Cluster ID for each short text.
1  // Step 1: Constructing word graph.
2  Initiate a multiset $V' = \emptyset$, a multiset $E' = \emptyset$, a set $SeedSet = \emptyset$ and a set $C = \emptyset$.
3  for $d \in D$ do
4      $V' = V' \cup \{w_i | w_i \in d\}$.
5      $E' = E' \cup \{(w, v) | w \in d, v \in d, w \neq v\}$.
6  end
7  $[V, \{f(w)\}] \leftarrow$ CountFrequency($V'$).
8  $[E, \{f(w, v)\}] \leftarrow$ CountFrequency($E'$).
9  $E = E \backslash \{(w, v) | f(w, v) < \gamma\}$.
10 Construct a node-weighted and edge-weighted graph $G$ based on $V$ and $E$.
11 //Step 2: Extracting topic representative term groups.
12 Sort nodes in $V$ in descending order according to node weight $f(w)$.
13 for node $w \in V$ do
14     Retrieve $w$'s neighbors in $G$ as set $N(w)$.
15     if ($f(w) \geq f(v), \forall\, v \in N(w)$ and $v \notin SeedSet$) then
16         Initiate a set $c_w = \emptyset$ to store seed term $w$ and its affiliated terms.
17         for $v \in N(w)$ do
18             if ($\frac{f(v)}{f(w)} \geq \delta$ and $\frac{f(w,v)}{f(v)} \geq \theta$) then
19                 Add $v$ into set $c_w$.
20             end
21         end
22         if $c_w\ != \emptyset$ then
23             Add $w$ into both set $SeedSet$ and set $c_w$.
24             Add $c_w$ into term group set $C$.
25         end
26     end
27 end
28 // Step 3: Clustering short text.
29 for $d \in D$ do
30     Infer cluster ID for $d$ according to Eq. (1).
31 end
```

where δ and θ are two thresholds. Rule 3 is also used to measure the *closeness* between a seed term w and its affiliated term v. Note that, the condition $\frac{f(v)}{f(w)}$ requires that v has a *relatively high frequency* relative to w, and the condition $\frac{f(w,v)}{f(v)}$ requires that v gives *sufficient supports* to w. For each seed term w, we use $A(w)$ to denote all the affiliated terms of w. Then, each seed term and its affiliated terms form up a *topic representative term group*, denoted as c. Through Rule 1 and Rule 3, the term in NEWG with lower term frequency or lower co-occurrence frequency with seed terms will not be chosen as representative terms. In this way, TRTD discards noise terms that have relatively lower term statistics.

Note that, if we randomly search seed terms in graph G, we may need to examine one node multiple times. For example in Figure 2 (b), based on the selection criteria in rule 2, both terms `black` and `nokia` will be extracted as seed terms. Suppose in current round, the node in examination is `nokia` but `black` has not been determined as a seed term, then the term `nokia` cannot be extracted as a seed term in this round, as its neighbor `black` has a higher frequency. Hence, `nokia` will be examined multiple times. To quickly find all seed terms, we examine all nodes in graph G in descending order in terms of node weight. In this way, all of the terms in G will be examined only once, which significantly decreases the computational cost. Lines 11–27 in **Algorithm 1** present the details of this step.

D. SHORT TEXT CLUSTER MEMBERSHIP ASSIGNMENT

The topic representative term groups are extracted from the short text corpus as core terms to represent short text clusters. Therefore, we can use them as virtual cluster centers to group short texts. Short texts are clustered into the same group if they share the same virtual closest cluster center.

Suppose that we have extracted K topic representative term groups, $C = \{c_1, c_2, \cdots, c_K\}$. We define the similarity between a short text and a keyword group by considering the length of their overlapped terms. For an arbitrary short text $d_i \in D$, $1 \leq i \leq m$, we assign the cluster index of d_i as k, if d_i has a larger number of common terms with topic representative term group c_k than any other groups. The following equation gives the criteria of cluster membership assignment for short texts:

$$l_i = \arg\max_{c_j \in C} |d_i \cap c_j|, \tag{1}$$

where $|d_i|$ denotes the number of terms in document d_i, $|d_i \cap c_j|$ denotes the number of common terms shared by d_i and c_j. $l_i \in [1, K]$ represents the cluster index of short text document d_i.

The overall process of TRTD for short text clustering is presented in **Algorithm 1**. In step 1 (at Lines 1-10), we calculate term statistics, such as $f(w), f(w, v)$ and $N(w)$, and construct the node-weighted and edge-weighted word graph G based on the term statistics. We use function CountFrequency shown in line 7-8 to calculate the corpus-level term frequency and term co-occurred frequency by adding all the terms and term pairs into two multisets (allowing for multiple instances), respectively. In step 2 (at Lines 11-27), we extract representative topic term groups. We infer the cluster label for short texts in step 3 (at Lines 28-31).

Note that, the computational cost of TRTD is spent on three steps. TRTD firstly extracts the statistical information of words, such as word frequency and word co-occurrence frequency and constructs node weighted and edge weighted word graph, with the time cost of $\mathcal{O}(|D| * (\frac{\bar{l}*(\bar{l}-1)}{2}))$, where $\bar{l}$ is the average length of short text documents, $|D|$ is the document number of the dataset. Then, TRTD discovers topic representative term groups based on NEWG with the complexity of $\mathcal{O}(|V| * \bar{N})$, where $|V|$ is the vocabulary size and $\bar{N}$ is the average effective neighboring terms for the seed terms. Finally, TRTD computes the distance between short texts and the discovered keyword groups to determine the cluster label of short texts with the complexity of $\mathcal{O}(|D| * K)$. The total time complexity of TRTD is $\mathcal{O}(|D| * (\frac{\bar{l}*(\bar{l}-1)}{2} + K) + |V| * \bar{N})$.

IV. EXPERIMENTS

In this section, we present our experimental setup in Section IV-A, discuss the optimal parameters of the proposed method in Section IV-B and evaluate the accuracy, effectiveness and efficiency for short text clustering in Section IV.C-E, respectively.

A. EXPERIMENTAL SETUP

1) DATASETS

We adopted two real-word short text datasets with ground truth cluster labels in our experiments. Table 2 shows the basic statistics of the datasets.

TABLE 2. Two short text datasets (Title and Tweet).

Dataset	Title	Tweet
Document size	56, 886	167,136
Vocabulary size	19, 120	70,423
Average length of documents	6.10	7.54
Cluster number	73	164
Maximum cluster size	22, 745	105,485
Minimum cluster size	50	50
Average cluster size	779	1, 019

Title Dataset: The *Title* dataset is a combination of news titles used in Yin and Wang [6] and those used in our previous work (Huang *et al.* [27]). The original new titles dataset provided Yin *et al.* was crawled on November 27, 2013 from Google News website. Huang *et al.* crawled the news titles published between July 1, 2013 and November 2, 2013 from the website InfoPig [2] and identified around 20 events from the news titles. Here, we combine the two datasets into a bigger news title dataset. We remove the clusters that contain less than 50 short texts. The combined *Title* dataset includes 56, 886 different news titles and each title averagely comprises 6.10 words. These titles have been grouped into 73 clusters and they have 19, 120 distinct words in the dataset. The maximum title cluster contains 22, 745 short texts. The minimum cluster contains 50 short text documents. The average cluster size is 779.

Tweet Dataset: The *Tweet* dataset is a combination of tweets provide by Yin *et al.* [28] and Yang and Leskovec [29]. Yin et al. collected tweets on Text REtrieval Conference (TREC)[3] in the 2011-2015 micro-blog tracks. We select the tweets provided by Yang et al. related to 16 different events happened in June 2009. To find the true historical events, we exploit the website[4] that records the major

[2]http://www.infopig.com/
[3]http://trec.nist.gov/data/microblog.html
[4]https://www.onthisday.com/

historical events happened in the world. Here, we combine the two tweet datasets into a bigger new tweet dataset. We remove the tweets clusters that contain less than 50 short texts. The combined *Tweet* dataset contains 167, 136 tweets and each tweet averagely comprises 7.54 words. These tweets have been grouped into 164 clusters and they have 70, 423 distinct words in the dataset. The maximum tweet cluster contains 105, 485 tweets. The minimum cluster has 50 tweets. The average cluster size is 1, 019.

Note that, the statistical information shown in Table 2 is collected after pre-processing original datasets by removing duplicated short text documents, converting letters into lowercase, removing stop words and words stemming.

2) COUNTERPART METHODS

We have compared the proposed TRTD with the following methods for short text clustering.

GSDMM [6] is a model-based clustering method for short texts. It assumes that each short text is generated from single latent topic and use Gibbs sampling technique to infer the topic index for each short text. Short texts belong to the same latent topic are grouped as a cluster.

BTM [9] is a probabilistic topic model for short texts. BTM infers the topics of short text corpus with the assumption that two co-occurrence words in a short text documents are generated from the same topic. BTM aims to overcome the sparsity of short texts by explicitly modeling the word co-occurrence patterns. Short texts that are generated by the same latent topic with the maximum probability are grouped as a cluster.

GPU-DMM [15] is a probabilistic topic model for short texts based on the Dirichlet Multinomial Mixture (DMM) model. GPU-DMM aims to ease the context sparsity of short texts by incorporating the recent neural network language model techniques to promote the semantically related words but rarely co-occurred words under the same topic with the generalized Polya urn (GPU) model. Short texts belong to the same latent topic are grouped as a cluster.

STC^2 [4] is a deep learning based clustering framework for short texts. It first adopts Convolutional Neural Networks to learn deep representation for short texts. After that, based on the new representations of short texts, K-Means is used to do the clustering task.

LDA [25] is a topic model based on assumptions that documents are generated by mixture latent topics in the corpus. Each document can be represented with a topic distribution vector which is inferred based on the words occurrence patterns. K-Means is adopted on the topic distribution representation for clustering task.

LSI [12] adopts singular value decomposition to factorize document-term matrix to identify patterns in the relationships between the document and the latent semantic space. Similar to LDA, the raw highly dimensional representation for documents in vector space model can be reduced into lower dimensional latent semantic space. K-Means is adopted on the latent semantic space representation for clustering task.

TextRank [16] is a graph-based ranking model for text processing. It can be used to find the keywords for a regular sized document. It constructs word graph with edge weight based word co-occurrence frequency. For clustering short text data, we adopt the cluster membership assignment strategies shown in section III-D.

We adopt the open source code for GSDMM, BTM, GPU-DMM, STC^2 and TextRank. For the implementation of LDA and LSI, we use the machine learning package: scikit-learn [30] and Gensim.[5] For BTM, GSDMM, GPU-DMM and LDA, they involve two main parameters: the topic number K and the Gibbs sampling iteration number I. We set K as the cluster numbers in the dataset and I as 500 for better accuracy. All the methods are run for 10 times to report the average clustering accuracy.

3) EVALUATION METRICS

As the adopted short text datasets include ground truth cluster labels, we adopt three popular external metrics: Adjusted Rand Index (ARI) [9], Adjusted Mutual Information (AMI) [31] and Normalized Mutual Information (NMI) [9] to evaluate the clustering accuracy. We assume that the ground truth cluster partition of the dataset is denoted as $\mathcal{C} = \{c_1, \cdots, c_J\}$, where c_i is the i-th cluster in the dataset. The predicted partition for the dataset is denoted as $\Omega = \{\omega_1, \cdots, \omega_K\}$, where ω_j is the j-th predicted cluster for the dataset. The metric are explained as follows:

a: ADJUSTED RAND INDEX (ARI)

The clustering process can be regard as a series of steps to decide the cluster labels of two short texts. If two short texts are originally in the same ground truth cluster and are predicted into the same cluster, or they are not in the same ground truth cluster but predicted into different clusters, then the decision is correct. Rand Index calculates the percentage of correct decisions. ARI is an improved version of Rand Index, which is defined as follows:

$$ARI(\Omega, \mathcal{C}) = \frac{\sum_{i,j}\binom{|\omega_i \cap c_j|}{2} - [\sum_i \binom{|\omega_i|}{2} \sum_j \binom{|c_j|}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{|\omega_i|}{2} + \sum_j \binom{|c_j|}{2}] - [\sum_i \binom{|\omega_i|}{2} \sum_j \binom{|c_j|}{2}]/\binom{n}{2}}, \quad (2)$$

b: ADJUSTED MUTUAL INFORMATION (AMI)

Mutual Information (MI) measures the percentage of same information sharing by two partitions. AMI improves MI for the fact that MI may become less accurate when the number of clustering partitions is big. Let $I(\Omega; \mathcal{C})$ denote the mutual information of the two predicted partition and ground truth partition for the dataset. $H(\Omega)$ and $H(\Omega)$ denote the entropy of Ω and $\mathcal{C}$. $E[I(\Omega; \mathcal{C})]$ denotes the expectation of $I(\Omega; \mathcal{C})$. AMI is defined as followings:

$$AMI(\Omega, \mathcal{C}) = \frac{I(\Omega; \mathcal{C}) - E[I(\Omega; \mathcal{C})]}{max[H(\Omega), H(\mathcal{C})] - E[I(\Omega; \mathcal{C})]} \quad (3)$$

[5]https://radimrehurek.com/gensim/

c: NORMALIZED MUTUAL INFORMATION (NMI)

NMI is designed as a compromise between the accuracy of the clustering and the total number of clusters. Similar to AMI, NMI is also an entropy-based metric that evaluates the amount of common information between two partitions:

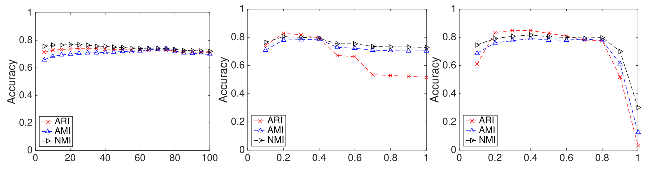
$$\mathrm{NMI}(\Omega, \mathcal{C}) = \frac{I(\Omega; \mathcal{C})}{[H(\Omega) + H(\mathcal{C})]/2} \tag{4}$$

The range of ARI, AMI and NMI is from 0 to 1, a larger value indicates a higher agreement between the ground truth partitions $\mathcal{C}$ and the predicted partitions Ω for the dataset.

B. OPTIMAL PARAMETER ANALYSIS FOR THE PROPOSED TRTD

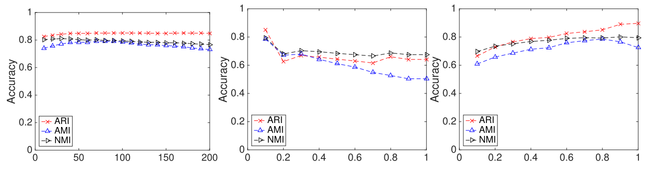
In this subsection, we study the optimal parameter settings of TRTD.

Figs. 3 and 4 show the clustering accuracy of TRTD with different parameter settings on Title and Tweet datasets, respectively. More specifically, Figs. 3 (a) and 4 (a) show the performance of TRTD changing with different γ setting. Here, we fix δ and θ with the best performance setting. As Tweet dataset is much larger than the remaining two datasets, therefore, the parameter γ on Tweet dataset is set from 10 to 200. As we can see, almost all of the three metrics show similar trend with the changing γ on the datasets. What is more, the performance of TRTD does not change too much with varying γ setting. This indicates that TRTD is insensitive to parameter γ.



(a) Parameter γ (b) Parameter δ (c) Parameter θ

FIGURE 3. TRTD parameter analysis for Title dataset.



(a) Parameter γ (b) Parameter δ (c) Parameter θ

FIGURE 4. TRTD parameter analysis for Tweet dataset.

Figs. 3 (b) and 4 (b) plot the performance of TRTD changing with δ from 0 to 1. We can see that all of the metrics have the similar trend when δ changes. Specifically, the performance of TRTD on Title and Tweet datasets show different trend when δ is less than 0.4. After δ is above 0.4, the performance of TRTD on the two datasets show very similar trend with the changing parameter δ.

Figs. 3 (c) and 4 (c) show the performance of TRTD changing with θ from 0 to 1. The experimental results on three metrics still show similar trend when θ is changed. We can see that the best θ for Title datasets is 0.5. For Tweet dataset, a higher θ setting shows better cluster performance. Analyzing the results of Figs. 3 and 4, we can obtained different optimal parameters for TRTD on the two datasets. The optimal parameters for Title are: $\gamma = 30$, $\delta = 0.2$ and $\theta = 0.5$ and for Tweet are: $\gamma = 30$, $\delta = 0.1$ and $\theta = 0.8$.

C. SHORT TEXT CLUSTERING ACCURACY ANALYSIS

In this subsection, we study the clustering accuracy of TRTD and the counterpart methods. Table 3 shows the clustering performance of all 8 methods on the two short text datasets.

TABLE 3. Short text clustering accuracy comparison between the proposed method and 7 counterpart methods on Title and Tweet datasets (The score with bold face indicates the best accuracy. The score with underline indicates the second best accuracy).

Methods	Metrics	Title	Tweet
TRTD	NMI	**0.804**	**0.810**
	ARI	**0.828**	**0.842**
	AMI	**0.781**	**0.771**
GSDMM	NMI	$\underline{0.723}$	$\underline{0.623}$
	ARI	0.315	$\underline{0.232}$
	AMI	$\underline{0.617}$	$\underline{0.509}$
BTM	NMI	0.654	0.542
	ARI	0.153	0.034
	AMI	0.516	0.339
GPU-DMM	NMI	0.722	0.620
	ARI	0.311	0.230
	AMI	0.617	0.506
STC^2	NMI	0.464	0.382
	ARI	0.063	0.011
	AMI	0.352	0.225
LDA	NMI	0.409	0.378
	ARI	0.085	0.016
	AMI	0.314	0.228
LSI	NMI	0.587	0.497
	ARI	0.165	0.066
	AMI	0.486	0.322
TextRank	NMI	0.674	-
	ARI	$\underline{0.755}$	-
	AMI	0.548	-

As we can see in Table 3, the overall trend is that TRTD achieves the best clustering performance on the two short text datasets. Among 7 counterpart methods, GSDMM is very competitive, it ranks the second in all three metrics on both Tweet and Title except in ARI on Title. TextRank ranks the second in ARI on the Title dataset. However, we can see that our TRTD performs far better than both GSDMM and TextRank, which achieves 0.804 (NMI), 0.828 (ARI) and 0.781 (AMI) on Title and 0.810 (NMI), 0.842 (ARI) and 0.771 (AMI) on Tweet. The proposed TRTD achieves very similar accuracy for both Title (regular short text) and Tweet (instant short text).

TABLE 4. The topic representative terms in the example short text topics.

Topic	Representative words
“rogen west”	rogen franco seth bound james kanye west video kim parody
“music gala”	prince william taylor swift bon jovi jon prayer gala sings
“packer rodgers”	packer viking rodgers lion flynn aaron green bay matt tie
“syria peace”	syria talk peace geneva syrian january iran conference set opposition
“alec baldwin”	baldwin alec msnbc gay slur late fired talk cancel joan

Considering the performance in three metrics on both datasets, the most competitive methods for TRTD are GSDMM and GPU-DMM. GSDMM and GPU-DMM achieve around 0.72 and 0.62 NMI accuracy on Title and Tweet datasets, respectively. As for ARI accuracy, GSDMM achieves around 0.232, which is slight better than GPU-DMM. For AMI accuracy, the two methods also show similar results, with 0.61 and 0.51 on two datasets, respectively. Both GSDMM and GPU-DMM adopt the Dirichlet mixture model to discover the topic index for short text corpus. Different from GSDMM, GPU-DMM incorporates the word semantic similarity using word embedding techniques. From the result, we can see that both methods do not perform well on large short text corpus.

STC^2 incorporates semantic information from word embedding techniques and adopts Convolutional Neural Network to learn deep representation for short texts. It achieves 0.46 and 0.38 NMI accuracy on the two datasets. But STC^2 shows worse ARI accuracy results, which are around 0.06 and 0.01 on the two datasets. LSI shows better NMI results than STC^2, which are around 0.59 and 0.50. The NMI results of LDA are around 0.41 and 0.38 on the two datasets, but LDA also shows worse ARI results. LDA is a topic model for regular length documents, it assumes that each document contains several different latent topics or semantics. This assumption may be not suitable for short texts due to their significant shortness and sparseness. Hence, LDA does not show better result compared with the other methods.

“-” in Table 3 indicates that we did not get results for TextRank as TextRank need lots of memory resources to process large corpus. TextRank achieves 0.674 (NMI), 0.755 (ARI) and 0.548 (AMI) accuracy on Title, which are much better than STC^2, LSI and LDA. This indicates that using keywords is promising to cluster short text corpus. But TextRank does not show better result compared with TRTD. TextRank extracts keywords but it can not determine if several keywords are from the same short text group. The results indicate that our proposed TRTD is better than TextRank for short text clustering.

D. EFFECTIVENESS ANALYSIS IN TOPIC TERM DISCOVERY

In this subsection, we analyze the effectiveness of topic representative terms discovered by TRTD, GSDMM, BTM, and GPU-DMM using typical examples. We choose 5 large topics (“rogen west”, “music gala”, “packer rodgers”, “syria peace”, and “alec baldwin”), which are supported by large numbers of short texts in the *Title* dataset. We adopt the top 10 most frequent terms in each cluster to represent the topic. Table 4 shows the ground truth and Tables 5-8 show the results of topic representative term groups discovered by TRTD, GSDMM, BTM and GPUDMM, respectively.

TABLE 5. The topic representative terms discovered by TRTD. The boldface terms are wrongly detected.

Topic	Representative words
"rogen west"	kanye west kim **kardashian** franco rogen bound james seth parody
“music gala”	taylor prince william swift bon jovi jon prayer gala sings
“packer rodgers”	packer viking rodgers lion green tie flynn matt **tolzien**
“syria peace”	talk syria peace geneva syrian iran january
“alec baldwin”	baldwin alec msnbc late gay slur **defends** fired cancel joan

TABLE 6. The topic representative terms discovered by GSDMM. The boldface terms are wrongly detected.

Topic	Representative words
“rogen west”	rogen franco bound seth kanye james west video kim parody
“music gala”	taylor prince william swift bon jovi jon prayer gala sings
“packer rodgers”	packer **patriot bronco** viking rodgers lion flynn **aaron** green matt
“syria peace”	syria peace **watkins** syrian geneva talk iran **lost-prophets assad jan**
“alec baldwin”	baldwin alec **oldboy** msnbc gay **spike lee** cancel slur fired

TABLE 7. The topic representative terms discovered by BTM. The boldface terms are wrongly detected.

Topic	Representative words
"rogen west"	rogen franco seth james bound kanye west video parody kim
"music gala"	taylor prince william swift bon jovi jon prayer gala sings
“packer rodgers”	packer viking flynn matt **tolzien** tie **game scott qb peterson**
“syria peace”	talk syria peace geneva syrian january **will** set conference **government**
“alec baldwin”	baldwin alec msnbc gay slur **late** talk **joan river** cancel

TABLE 8. The topic representative terms discovered by GPU-DMM. The boldface terms are wrongly detected.

Topic	Representative words
"rogen west"	kanye rogen franco bound seth james west kim video parody
"music gala "	taylor william prince swift bon jovi jon prayer gala sings
“packer rodgers”	packer viking rodgers lion flynn aaron green bay matt tie
“syria peace”	talk syria peace geneva **security karzai** syrian **deal** january **afghan** 9
“alec baldwin”	**kobe bryant** baldwin **lakers** alec msnbc **extension contract year** gay

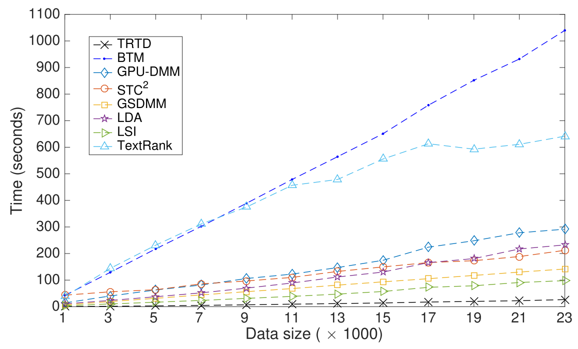


FIGURE 5. Time cost of different methods.

Compared with the ground truth in Table 4, we can observe that TRTD (as shown in Table 5) can discover more accurate top frequent topic terms than other methods (as shown in Tables 5-8. Taking the topic "`packer rodgers`" as an example, TRTD can discover 9 out of 10 most representative words for the cluster "`packer rodgers`", while BTM misses 5 and GSDMM misses 3. GPU-DMM is accurate to discover the top frequent representative terms for topic "`rogen west`","`music gala`" and "`packer rodgers`", but fails to discover the other two topics: "`syria peace`" and "`alec baldwin`". For example, the topic "`alec baldwin`" is about the scandal news of movie star "Alec Baldwin", but GPU-DMM also extracts the basketball star "Kobe Bryant" as the representative for this topic. The proposed TRTD focuses on discovering the most significant term groups for short text clustering by considering the closeness relations of two terms in the word network. Therefore, TRTD can discover more accurate top frequent terms as the representative terms and filters trivial terms at the same time.

E. SHORT TEXT CLUSTERING EFFICIENCY ANALYSIS

In this subsection, we demonstrate the efficiency of TRTD. All the experiments were conducted on a Linux Server with 2.30 GHz CPU and 64GB memory. TRTD, GPU-DMM, STC^2, LDA, LSI and TextRank were implemented in Python. BTM and GSDMM were implemented in Java. For the probabilistic model based methods like BTM, GSDMM, GPU-DMM and LDA, we set their iteration number with 100 as a lower iteration setting will lead to inaccurate clustering result.

Fig. 5 shows the execution time of different methods changing with the increased data sizes. As we can see, the time cost for these methods are approximately linear to the size of the dataset, but their running speed is different. TRTD costs less execution time and is apparently faster than other methods. On the contrary, the counterpart methods show worse efficiency compared with TRTD. Taking BTM as an example, BTM need to repetitively sample topic for each bi-term of the dataset multiple times and show the worst time efficiency among all the methods. TextRank show the second worst time efficiency. GSDMM and LSI show similar time cost within the increasing of dataset size.

V. CONCLUSION

We proposed a topic representative terms discovery (TRTD) method for short text clustering in this paper. TRTD exploits a node-weighted and edge-weighted word graph to find groups of significant terms that are closely bounded with each other as a topic representative term group, resolves the noisy and insufficient topic term discovery problem in the previous methods. TRTD also addresses the issues of sparsity and noisy in short texts clustering. Extensive experiments on real-world datasets show that our approach outperforms 7 counterpart methods in terms of accuracy, effectiveness and efficiency. Some future directions can be explored based on our proposed TRTD. As TRTD is efficient and effective, one of the potential research aspects in the future is to extend TRTD into clustering short text streams since short texts are often continuously generated.

REFERENCES

[1] F. Atefeh and W. Khreich, "A survey of techniques for event detection in Twitter," *Comput. Intell.*, vol. 31, no. 1, pp. 132–164, 2015.

[2] L. Wu and H. Liu, "Tracing fake-news footprints: Characterizing social media messages by how they propagate," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, 2018, pp. 637–645.

[3] G. Paltoglou and M. Thelwall, "Twitter, myspace, digg: Unsupervised sentiment analysis in social media," *ACM Trans. Intell. Syst. Technol.*, vol. 3, no. 4, p. 66, 2012.

[4] J. Xu , B. Xu, P. Wang, S. Zheng, G. Tian, and J. Zhao, "Self-taught convolutional neural networks for short text clustering," *Neural Netw.*, vol. 88, pp. 22–31, Apr. 2017.

[5] C. Jia, M. B. Carson, X. Wang, and J. Yu, "Concept decompositions for short text clustering by identifying word communities," *Pattern Recognit.*, vol. 76, pp. 691–703, Apr. 2018.

[6] J. Yin and J. Wang, "A Dirichlet multinomial mixture model-based approach for short text clustering," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 233–242.

[7] S. Banerjee, K. Ramanathan, and A. Gupta, "Clustering short texts using wikipedia," in *Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2007, pp. 787–788.

[8] O. Jin, N. N. Liu, K. Zhao, Y. Yu, and Q. Yang, "Transferring topical knowledge from auxiliary long texts for short text clustering," in *Proc. 20th ACM Int. Conf. Inf. Knowl. Manage.*, 2011, pp. 775–784.

[9] X. Yan, J. Guo, Y. Lan, and X. Cheng, "A biterm topic model for short texts," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 1445–1456.

[10] X. Ni, X. Quan, Z. Lu, L. Wenyin, and B. Hua, "Short text clustering by finding core terms," *Knowl. Inf. Syst.*, vol. 27, no. 3, pp. 345–365, 2011.

[11] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[12] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *J. Amer. Soc. Inf. Sci.*, vol. 41, no. 6, pp. 391–407, 1990.

[13] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

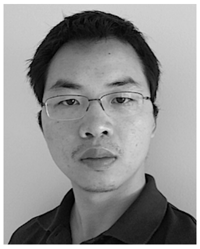
[14] Y. Kim, "Convolutional neural networks for sentence classification," 2014, *arXiv:1408.5882*. [Online]. Available: https://arxiv.org/abs/1408.5882

[15] C. Li, H. Wang, Z. Zhang, A. Sun, and Z. Ma, "Topic modeling for short texts with auxiliary word embeddings," in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2016, pp. 165–174.

[16] R. Mihalcea and P. Tarau, "TextRank: Bringing order into text," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2004, pp. 1–8.

[17] Y. Li, C. Jia, and J. Yu, "A parameter-free community detection method based on centrality and dispersion of nodes in complex networks," *Phys. A, Stat. Mech. Appl.*, vol. 438, pp. 321–334, Nov. 2015.

[18] S. Jinarat, B. Manaskasemsak, and A. Rungsawang, "Short text clustering based on word semantic graph with word embedding model," in *Proc. Joint 10th Int. Conf. Soft Comput. Intell. Syst. (SCIS)*, Dec. 2018, pp. 1427–1432.
[19] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manage.*, vol. 24, no. 5, pp. 513–523, 1988.
[20] K. Liu, A. Bellet, and F. Sha, "Similarity learning for high-dimensional sparse data," in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)*, 2015, pp. 1–10.
[21] J. Tang, X. Wang, H. Gao, X. Hu, and H. Liu, "Enriching short text representation in microblog for clustering," *Frontiers Comput. Sci.*, vol. 6, no. 1, pp. 88–101, 2012.
[22] C. T. Zheng, C. Liu, and H. S. Wong, "Corpus-based topic diffusion for short text clustering," *Neurocomputing*, vol. 275, pp. 2444–2458, Jan. 2018.
[23] X. Huang, Y. M. Ye, X. L. Du, and S. Deng, "Short text clustering with expanding keywords through concept graph," *J. Comput. Inf. Syst.*, vol. 9, no. 21, pp. 8649–8657, 2013.
[24] M. Kozlowski and H. Rybinski, "Clustering of semantically enriched short texts," *J. Intell. Inf. Syst.*, pp. 1–24, Dec. 2018.
[25] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.
[26] J. Qiang, Y. Li, Y. Yuan, and X. Wu, "Short text clustering based on Pitman–Yor process mixture model," *Appl. Intell.*, vol. 48, no. 7, pp. 1802–1812, 2018.
[27] 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.
[28] J. Yin, D. Chao, Z. Liu, W. Zhang, X. Yu, and J. Wang, "Model-based clustering of short text streams," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, New York, NY, USA, 2018, pp. 2634–2642.
[29] J. Yang and J. Leskovec, "Patterns of temporal variation in online media," in *Proc. 4th ACM Int. Conf. Web Search Data Mining (WSDM)*, New York, NY, USA, 2011, pp. 177–186.
[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.
[31] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res.*, vol. 11, pp. 2837–2854, Jan. 2010.



SHUIQIAO YANG received the B.S. degree from Shanghai Ocean University, China, and the M.S. degree from Zhejiang University, China. He is currently pursuing the Ph.D. degree with the School of Information Technology, Deakin University, Australia. His research interests include data analysis, text mining, and pattern recognition.

GUANGYAN HUANG received the Ph.D. degree in computer science from Victoria University, in 2012. She was with the Chinese Academy of Sciences, from 2003 to 2009, and visited the Platforms and Devices Centre, Microsoft Research Asia, in 2006. She is currently a Senior Lecturer with the School of Information Technology, Deakin University. She is a Chief Investigator of two ARC Discovery Projects. She is a Principal Supervisor of three Ph.D. students. She has over 80 publications mainly in data mining, the IoT/sensor networks, text analytics, image/video processing, spatiotemporal database, and intelligent threat modeling. She was a recipient of an ARC Discovery Early Career Researcher Awards (DECRA) Fellowship.

BORUI CAI received the bachelor's and master's degrees in engineering from Beihang University (BUAA), in 2010 and 2013, respectively. He is currently pursuing the Ph.D. degree with the School of Information, Deakin University. He was with the Chinese Academy of Sciences, from 2013 to 2016, and was with Xilinx, Inc., in 2017. His research interests include privacy protection, time series analysis, and pattern recognition.