Applied Intelligence (2022) 52:4651–4662
https://doi.org/10.1007/s10489-021-02263-z

Dynamic clustering for short text stream based on Dirichlet process

Wanyin Xu[1] · Yun Li[1] · Jipeng Qiang[1]

Accepted: 4 February 2021 / Published online: 26 July 2021

Abstract
Due to the explosive growth of short text on various social media platforms, short text stream clustering has become an increasingly prominent issue. Unlike traditional text streams, short text stream data present the following characteristics: short length, weak signal, high volume, high velocity, topic drift, etc. Existing methods cannot simultaneously address two major problems very well: inferring the number of topics and topic drift. Therefore, we propose a dynamic clustering algorithm for short text streams based on the Dirichlet process (DCSS), which can automatically learn the number of topics in documents and solve the topic drift problem of short text streams. To solve the sparsity problem of short texts, DCSS considers the correlation of the topic distribution at neighbouring time points and uses the inferred topic distribution of past documents as a prior of the topic distribution at the current moment while simultaneously allowing newly streamed documents to change the posterior distribution of topics. We conduct experiments on two widely used datasets, and the results show that DCSS outperforms existing methods and has better stability.

Keywords Short text stream · Dirichlet process · Dynamic clustering · Topic drift

1 Introduction

Short texts are prevalent on the Web, including on traditional websites, e.g., news titles and search snippets, and emerging social media, e.g., microblogs and tweets. In recent years, these data have swept the world at an alarming rate, and have produced large quantities of data streams, also called short text streams. Short text stream clustering [1] is challenging due to the inherent characteristics of short text steams such as short length, weak signal and high ambiguity of each short text, and the explosive growth and popularity of short textual content.

Considering the characteristics of short text streams, it is difficult to apply traditional short text clustering approaches to model building because of the following two challenges: 1) insufficient information or statistical signals [2] in short text streams to make the analysis meaningful; and 2) topic distributions change with time [3], with previously salient topics "fading-off" and vice versa.

To address these challenges, Liang et al. [4] proposed a dynamic clustering topic model (DCT), that uses the relevance of topics between different time points to alleviate the sparsity problem of the short text stream. However, DCT requires manual specification of the number of topics, which is a major limitation when addressing real data streams. Later, Yin et al. [5] proposed a short text stream clustering algorithm based on the Dirichlet process multinomial mixture model, called MStream, which can handle concept drift.

However, MStream does not consider the correlation of the topic distribution at neighbouring time points, and ignores the fact that documents with similar time points might have a higher probability of belonging to the same topic, that is, the phenomenon that "the discussion of events from the previous day may continue to the next day".

Therefore, to better solve the problem related to the sparsity and topic drift of the short text stream, we propose dynamic clustering for the short text stream based on the Dirichlet process [6], called DCSS. Our main contributions in this paper are as follows:

– DCSS can detect topic drift in short text streams. DCSS uses the Dirichlet process to initialize the topic of each

✉ Yun Li
liyun@yzu.edu.cn

Wanyin Xu
xuwany@yeah.net

Jipeng Qiang
jpqiang@yzu.edu.cn

1 Department of Software Engineering, Yangzhou University, Yangzhou, China

text and determine whether to add topics by calculating the probability of existing topics and potential new topics so that there is no need to initialize the number of topics in advance.
– DCSS can alleviate the sparsity problem of the short text stream. DCSS infers the hidden topic of the current short text based not only on the content of the text but also the topic distribution of the previous time point as prior information.
– DCSS is more efficient than baseline methods. We compare the existing algorithms on public datasets, and verify that DCSS outperforms the state-of-the-art baselines. The code to reproduce the results is available at https://github.com/SilverXWY/DCSS.

The following sections are organized as follows: Section 2 describes the related work; Section 3 details the formulation and procedure of DCSS; Section 4 presents the experimental results; Section 5 summarizes the work of this paper.

2 Related work

Two main types of methods are used to analyse text streams: text stream clustering based on similarity and text stream clustering based on topic models.

2.1 Text stream clustering based on similarity

Most text stream clustering methods based on similarity use vector space models to represent documents, and calculate the similarity between documents or clusters.

CluStream [7] is one of the most classic stream clustering methods that consists of online micro-clustering and offline macro-clustering. CluStream uses a pyramid time frame to store micro-clusters at different times in the past for future analysis. DenStream [8] combines micro-clustering with the density-estimation process of stream clustering to perform any form of data cluster and handle outliers. Yoo et al. [9] proposed a streaming spectral clustering method that maintains the approximation of the normalized Laplace operator for data streams over time and effectively updates the Laplace transformed eigenvectors as streams.

Zhong et al. [10] proposed an efficient text stream clustering algorithm built upon the well-known winner-take-all competitive learning that updates cluster centres online. Aggarwal and Yu [11] proposed a text and categorical data stream clustering method that summarizes data streams into fine-grained clusters. Shou et al. [12] proposed a prototype of a persistent summary for Twitter text streaming, called Sumblr which compresses tweets into tweet feature vectors (TCVs) and processes them online. Kalogeratos et al. [13] proposed a method for clustering text streams using burst word information. This approach makes use of the fact that most of the important documents in a topic were published during the eruption of the term "main".

The limitation of similarity-based clustering methods for text streams is the need to manually select a similarity threshold to determine whether documents are assigned to a new cluster, and the fact that there is no correlation between different time points.

2.2 Text stream clustering based on topic model

The topic model is the most typical unsupervised text clustering model. It assumes that text is generated through the process of "Select a topic with a certain probability and select a word with a certain probability from this topic". Generally, parameters are estimated by Gibbs sampling [14] or the EM algorithm [15].

Traditional static topic models are based on an entire corpus and cannot be applied directly to text streams. Therefore, many extended models for LDA [16], such as the dynamic topic model (DTM [17]), topic over time model (TOT [18]), dynamic mixture model (DMM [19]), online latent Dirichlet allocation model (OnlineLDA [20]), topic tracking model (TTM [21]), streaming latent Dirichlet allocation model (S-LDA [22]), and Dirichlet mixture model with feature partition (DPMFP [23]), have been proposed to handle text streams. These models have been applied to topic mining for long text streams, following the idea that words in a document may come from different topics. When the document is sufficiently long, the topic of an article may indeed be composed of words from unique topics, but when considering short text streams, this idea is contrary to reality.

Short texts such as microblogs often have only one or two sentences, and the main words often belong to the same topic. If the topic model generates each word from different topics, it will greatly affect the clustering performance and computation speed. To apply the topic model to short text streams, Yin et al. proposed a dynamic GSDMM [24] model that considers that all the words in a text as belonging to a single topic and effectively solves the sparsity problem of short text. The DCT [4] model proposed by Liang et al. assumes that topics at the previous time may have a guiding effect on topics at a later time. Therefore, the dependency relationship of the topic distribution between different time points is introduced, and the topic distribution at the previous time point is taken as a prior of the topic distribution of documents at the next moment. The oBTM [25] model proposed by Cheng et al. assumes that each bi-term in a text belongs to the same topic, which makes the grouping of document topics more realistic, and uses time slice to process the text stream. All text in a time slice can be iterated multiple times. After processing all the text in a time slice, the topic information of the bi-term is output to update the model parameters.

Although the above algorithm can infer the number of topics in a text stream during the iteration process, the number of topics must be specified in advance. If the number of topics specified manually is too large or too small, the clustering time and results can be affected substantially. Therefore, the TDMP [26] model proposed by Ahmed et al. based on the Dirichlet process automatically determines the number of topics to address the fact that the number of topics in real streaming documents is not known beforehand. However, the method needs the entire sequence of text streams. The MStream [5] algorithm proposed by Yin et al. uses a forget rule to update the topic distribution based on the Dirichlet process to solve the problem of topic drift. The NPMM [27] model is a recently introduced model that uses word embeddings to eliminate a cluster generating parameter from the model. The above algorithms address the problem of specifying the number of topics and topic drift, but ignore the fact that documents at neighbouring time points may have a higher probability of belonging to the same topic. For example, there is a good probability that text with the word "mask" will belong to the same topic of "epidemic" as subsequent text with the word "Thunder God Mountain" during the COVID-19 pandemic.

Therefore, the algorithm in this paper uses the Dirichlet process to automatically obtain the number of topics and adopts temporal dependency to fully consider the topic relevance of the text stream at neighbouring time points. DCSS updates the topic at each point in time and only saves the text statistics of the previous moment, which addresses topic drift without occupying excessive space resources.

3 Proposed approach

We propose dynamic clustering for short text streams based on the Dirichlet process (DCSS). DCSS can automatically generate new topics based on the Dirichlet process without requiring the number of topics as input. To account for topic drift, a fundamental challenge in short text streams, we adopt topic feature tuples to update topics at each time, and delete outdated topic information. The update of topic feature tuples plays a major role in handling the dynamic problem of text streams. In the clustering task at each time point, according to the temporal dependency strategy, we refer to the statistical information of the text clustering result at the previous time point, which in large part solves the sparsity problem of short texts. When processing the streaming text at each time point, Dirichlet process clustering is applied to obtain a topic distribution Θ and word distribution Φ at the current time. The topic distribution of text in reality may evolve over time. Since the probability that the text belongs to a topic can be inferred from the topic distribution and the word distribution, we discuss the text generation process based on the Dirichlet process in detail and describe how to adopt the temporal dependency in this process.

3.1 Generative process

The Dirichlet process [6] is a stochastic process, most commonly used as a prior for mixture models, which is widely used in nonparametric Bayesian models. In the generative process, topics and words are drawn from the multinomial distributions of the mixture model, and the Dirichlet distribution is the prior of these multinomial distributions.

Table 1 summarizes the main notation used in our model.

We let $\Theta_t = \{\theta_{t,z}\}_{z=1}^{Z}$ represent the topic distribution of the incoming text stream at time t, where $\theta_{t,z} = P(z|t) > 0$, $\sum_{z=1}^{Z} \theta_{t,z} = 1$, and Z is the number of topics. $\Phi_t = \{\phi_{t,z}\}_{z=1}^{Z}$ represents the distribution of words in the text stream at time t, where $\phi_{t,z} = \{\varphi_{t,z,w}\}_{w=1}^{V}$ is the multinomial distribution of all words corresponding to topic z, V represents the size of the current time vocabulary $\mathbf{V}$, and $\varphi_{t,z,w} = P(w|t,z) > 0$, $\sum_{w=1}^{V} \varphi_{t,z,w} = 1$. The idea of processing static text is that the topic distribution at the current time is independent of the past distribution, so the Dirichlet prior of the topic distribution of each document is determined by only the static hyperparameter α.

$$P(\Theta_t|\alpha) \propto \prod_{z=1}^{Z} \theta_{t,z}^{\alpha} \tag{1}$$

Table 1 Main notations

d	document
z	topic
t	time point
w	word
z_d	topic assignment for document d
$m_{t,k}$	number of texts in topic k at time t
$n_{t,k}^{w}$	number of occurrences of word w in topic k at time t
N_d	number of words in document d
N_d^{w}	number of occurrences of word w in document d
B_t	number of documents arrived at time t
V	total number of currently recorded words
Θ_t	topic distribution at time t
Φ_t	word distribution at time t

At the same time, the Dirichlet prior to the word distribution $\phi_{t,z}$ of each topic is determined by only the hyperparameter β.

$$P(\phi_{t,z}|\beta) \propto \prod_{w=1}^{V} \varphi_{t,z,w}^{\beta} \tag{2}$$

Because of the higher probability that texts of neighbouring time points in the text stream belong to the same topic, we introduce time dependence, which makes the Dirichlet prior also depend on the topic distribution and word distribution of the previous time point. First, according to the parameter α and the topic distribution Θ_{t-1} at the previous time point, the topic distribution Θ_t at the current time point is generated, and the topic of the text is extracted. Then, all the words corresponding to the current text are extracted according to the topic. The distribution of words over the topic is generated by the parameter β and the word distribution Φ_{t-1} at the previous time point.

Since the topic distribution $\theta_{t,z}$ at each time point is related to $m_{t,z}$ which is the number of documents belonging to topic z, and the distribution of words over topic $\phi_{t,z}$ is related to $n_{t,z}^{w}$ which is the frequency of words w occurring in topic z, we change (1) and (2) to (3) and (4), respectively.

$$P(\Theta_t|\Theta_{t-1},\alpha) \propto \prod_{z=1}^{Z} \theta_{t,z}^{\alpha m_{t-1,z}} \tag{3}$$

$$P(\phi_{t,z}|\phi_{t-1,z},\beta) \propto \prod_{w=1}^{V} \varphi_{t,z,w}^{\beta n_{t-1,z}^{w}} \tag{4}$$

Since the probability of the text topic calculated at the current time point is related to that of the previous time point, the sparsity problem of short text is alleviated by providing more text information. However, the problem that topic models often need to fix the number of topics K has not yet been solved. Therefore, in the text generative process of our proposed algorithm, new topics are acquired dynamically based on the Dirichlet process, and the number of topics K is initially set to 0. The graphical representation of DCSS is illustrated in Fig. 1, and the parameterization is as follows:

$$\begin{aligned}
&\Theta_t \sim Dir(\alpha m_{t-1,z})\\
&\phi_{t,z}|\phi_{t-1,z},\beta \sim Dir(\beta n_{t-1,z}^{w}) \quad z = 1,\cdots,\infty\\
&z_d \sim Mult(\Theta_t) \quad d = 1,\cdots,B_t\\
&d|z_d,\{\phi_{t,z}\}_{z=1}^{\infty} \sim p(d|\phi_{t,z_d})
\end{aligned}$$

where Dir is a Dirichlet distribution, $Mult$ is a multinomial distribution, z_d represents the topic assigned to text d, and B_t represents the number of texts arriving at time t. Note that the number of topics in the text generative process is not fixed, as shown in Fig. 1. That is, in the Dirichlet process, there is no need to initialize the number of topics in advance. The Chinese restaurant process (CRP) [28] is a popular way to explain this process. Suppose a restaurant has an infinite number of tables and initially there is no one in the restaurant. The first customer enters and chooses the first table, and the next customer chooses either the $k_t h$ table with n_k people with probability of $\frac{n_k}{\alpha+n-1}$ or picks an empty table with probability of $\frac{\alpha}{\alpha+n-1}$. Therefore, at each moment, new text is dynamically assigned the topic, similarly to the CRP.

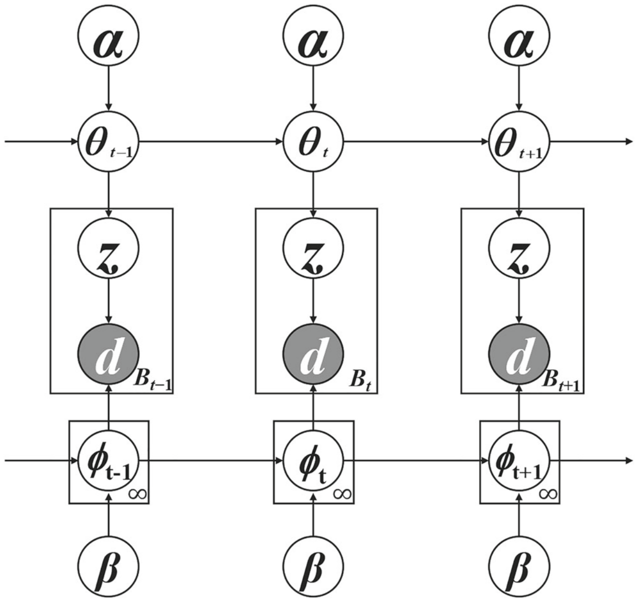

Fig. 1 The graphical representation of DCSS

3.2 Model formulation

This section presents a brief discussion of the formulation of DCSS.

Defining the relationship between documents and clusters is the most crucial task when addressing the text stream clustering problem. Similarity-based stream clustering methods use metrics such as cosine similarity to define the similarity between a document and a cluster. If the dissimilarity between existing clusters and newly arriving text exceeds a threshold, a new cluster is created; however, the similarity threshold is very difficult to define manually.

In contrast, based on the parameter estimation of the topic distribution Θ_t and word distribution $\phi_{t,z}$, we can calculate the probability that the text belongs to each existing topic. Moreover, we can calculate the probability that the text belongs to a new topic and finally choose the topic with the largest probability. Since each text has a possibility of being assigned to a new topic, DCSS can solve the topic drift problem.

At time t, the text d is the observable known variable, α and β are given prior parameters, and the topic z is hidden variable. According to the graphical model of DCSS, we define the joint distribution of text topics in (5):

$$\begin{aligned}
&P(B_t, Z_t|\Phi_{t-1}, \Theta_{t-1}, \alpha, \beta) = P(B_t|Z_t, \Phi_{t-1}, \beta)P(Z_t|\Theta_{t-1}, \alpha)\\
&= \int P(B_t|Z_t, \Phi_t)P(\Phi_t|\Phi_{t-1}, \beta)d\Phi_t \int P(Z_t|\Theta_t)P(\Theta_t|\Theta_{t-1}, \alpha)d\Theta_t\\
&= \int \prod_{d=1}^{|B_t|}\prod_{w=1}^{N_d} P(w, d|\phi^w_{t,z_d}) \prod_{z=1}^{K} P(\phi_{t,z}|\phi_{t-1,z}, \beta)d\Phi_t\\
&\quad\times \int \prod_{d}^{|B_t|} P(z_d|\theta_t)P(\theta_t|\theta_{t-1}, \alpha)d\Theta_t\\
&= \int \prod_{z=1}^{K}\prod_{w=1}^{V} \phi_{t,z,w}^{n^w_{t,z}} \prod_{z=1}^{K} P(\phi_{t,z}|\phi_{t-1,z}, \beta)d\Phi_t\\
&\quad\times \int \prod_{d}^{|B_t|} P(z_d|\theta_t)P(\theta_t|\theta_{t-1}, \alpha)d\Theta_t
\end{aligned} \tag{5}$$

where B_t represents the number of documents arriving at time t and Z_t represents the topic assignments for processed documents in B_t. We account for the temporal dependency by adding the topic distribution Θ_{t-1} and the word distribution over the topic Φ_{t-1} at the previous time to the joint distribution; that is, we use the inferred past document topic distribution as the prior of the topic distribution of documents at the current moment.

However, it is intractable to integrate out $\phi_{t,z}$ and Θ_t directly. Therefore, we employ collapsed Gibbs sample [14] for approximate inference and adopt a conjugate prior (Dirichlet) for the multinomial distributions, thus we can easily calculate the conditional distribution. Applying the chain rule, we can obtain the conditional probability according to (3) and (4):

$$\begin{aligned}
&P(z_d = k|Z_{\neg d}, B_t, \Phi_{t-1}, \Theta_{t-1}, \alpha, \beta) = \frac{P(Z_t, B_t|\Phi_{t-1}, \Theta_{t-1}, \alpha, \beta)}{P(Z_{t,\neg d}, B_t|\Phi_{t-1}, \Theta_{t-1}, \alpha, \beta)}\\
&\propto \frac{P(Z_t, B_t|\Phi_{t-1}, \Theta_{t-1}, \alpha, \beta)}{P(Z_{t,\neg d}, B_t|\Phi_{t-1}, \Theta_{t-1}, \alpha, \beta)}\\
&= \prod_{z=1}^{K} \frac{\prod_{w=1}^{V} \Gamma(n^w_{t,z} + \beta n^w_{t-1,z})}{\Gamma(\sum_w^V n^w_{t,z} + \beta n^w_{t-1,z})} \frac{\prod_{z=1}^{K} \Gamma(m_{t,z} + \alpha m_{t-1,z})}{\Gamma(\sum_{z=1}^{K} m_{t,z} + \alpha m_{t-1,z})}\\
&\Bigg/ \prod_{z=1}^{K} \frac{\prod_{w=1}^{V} \Gamma(n^w_{t,z,\neg d} + \beta n^w_{t-1,z})}{\Gamma(\sum_w^V n^w_{t,z,\neg d} + \beta n^w_{t-1,z})} \frac{\prod_{z=1}^{K} \Gamma(m_{t,z,\neg d} + \alpha m_{t-1,z})}{\Gamma(\sum_{z=1}^{K} m_{t,z,\neg d} + \alpha m_{t-1,z})}
\end{aligned} \tag{6}$$

Because document d is associated with its own topic z, and $\Gamma(x) = (x-1)\Gamma(x-1)$, $\Gamma(x+m) = \prod_{i=1}^{m}(x+i-1)\Gamma(x)$, we can simplify the conditional probability in (6). As a result, the derived equation for calculating the probability of a document d choosing existing topic k at time t is given in (7):

$$\begin{aligned}
&p(z_d = k|Z_{t,\neg d}, B_t, \Phi_{t-1}, \Theta_{t-1}, \alpha, \beta) \propto \frac{m_{t,k,\neg d} + \alpha m_{t-1,k}}{B_t + \alpha B_t + \alpha B_{t\text{ - }1}\text{ - }1} \cdot\\
&\frac{\prod_{w\in d}\prod_{j=1}^{N_d^w} (n^w_{t,k,\neg d} + \beta n^w_{t-1,k} + \beta + j - 1)}{\prod_{i=1}^{N_d} (n_{t,k,\neg d} + \beta n_{t-1,k} + V\beta + i - 1)}
\end{aligned} \tag{7}$$

where z_d represents the topic selected by text d, $m_{t,k}$ represents the number of texts in topic k at time t, $n^w_{t,k}$ represents the frequency of word w occurring in topic k at time t, $n_{t,k}$ represents the number of words in topic k at time t, B_t represents the number of texts arriving at time t, all "$t-1$" items are parameters corresponding to the previous time point, and V represents the size of the vocabulary of the currently recorded documents.

By following the Dirichlet process for infinite number of clusters, the probability of text d choosing a new topic $K+1$ is:

$$\begin{aligned}
&p(z_d = K+1|Z_{t,\neg d}, B_t, \Phi_{t-1}, \Theta_{t-1}, \alpha, \beta) \propto \frac{\alpha B_t}{B_t + \alpha B_t + \alpha B_{t\text{ - }1}\text{ - }1} \cdot\\
&\frac{\prod_{w\in d}\prod_{j=1}^{N_d^w} (\beta + j - 1)}{\prod_{i=1}^{N_d} (V\beta + i - 1)}
\end{aligned} \tag{8}$$

where K is the number of existing topics. Notably, when initializing the text at the first moment, there is no $t-1$ time point, so the corresponding parameter at time $t-1$ is 0.

The first term of (7) and (8) represents the completeness of the cluster, whereas, α is the concentration parameter of the model. A new document has a higher probability of choosing a topic with more documents, so the number of topics is limited to a certain number. The second term defines the homogeneity between a cluster and a document, and β is the pseudo weight of similar words in a cluster. When a topic has more documents that share same words with document d, the second part will be larger, and document d will be more likely to belong to that topic.

3.3 DCSS algorithm

To address the dynamic nature of text streams, DCSS adopts a strategy of updating topics at each point in time, assuming that the streaming texts at each moment can be clustered and iterated multiple times. Since the entire corpus is no longer uploaded statically, we need to define a feature tuple $\{m_{t,z}, n_{t,z}, n^w_{t,z}\}$ for cluster z corresponding to each topic at the current time, where $m_{t,z}$ represents the number of texts in topic z, and $n_{t,z}$ represents the number of words in topic z, $n^w_{t,z}$ represents the frequency of word w occurring in topic z at time t. The feature tuple of each topic z adds or deletes the information of a text to prepare for calculating the topic probability of the text. We only save feature tuples of topics in one moment, so that DCSS does not occupy excessive space resources. Therefore, after processing all the documents at the current time, we delete the feature tuples of the previous moment, and take the feature tuples of the current time as the prior for the next time point, and the outdated topics are deleted in this process.

Algorithm 1: DCSS(t time point)

```
Input: Feature tuples $\{m_{t-1,z}, n_{t-1,z}, n^w_{t-1,z}\}$ of all
       topics at the last time point;          Text
       stream $D_t = \{d_{t,1}, d_{t,2}, \ldots, d_{t,B_t}\}$ at the current
       time point.
Output: Feature tuples $\{m_{t,z}, n_{t,z}, n^w_{t,z}\}$ of all topics at
        the current time point;          Topics
        $Z_t = \{z_{d_{t,1}}, z_{d_{t,2}}, \ldots, z_{d_{t,B_t}}\}$ corresponding to
        the text stream at the current time point.
1  for $d = d_{t,1}$ to $d = d_{t,B_t}$ do
2      Calculate the probability of text $d$ belonging to
       each of the $K$ existing topics and a new topic. ;
       // According to (7) and (8).
3      According to the $K + 1$ probability, sample the
       topic $z$ of the text $d$.
4      if $z = K + 1$ then
5          $K = K + 1$
6          $m_{t,K} = \{\}$ ; $n_{t,K} = \{\}$ ; $n^w_{t,K}\{\}$;      // If
           text $d$ belongs to a new topic,
           initialize the feature tuple of
           the topic.
7      $m_{t,z} = m_{t,z} + 1$; $n_{t,z} = n_{t,z} + N_d$
8      for each word $w \in d$ do
9          $n^w_{t,z} = n^w_{t,z} + N^w_d$
10 for $iter = 1$ to $I$ do
11     for $d = d_{t,1}$ to $d = d_{t,B_t}$ do
12         Record the topic of the current text $d : z = z_d$
13         $m_{t,z} = m_{t,z} - 1$; $n_{t,z} = n_{t,z} - N_d$
14         for each word $w \in d$ do
15             $n^w_{t,z} = n^w_{t,z} - N^w_d$
16         Calculate the probability of text $d$ belonging to
           each of the $K$ existing topics and a new topic.
17         if $z = K + 1$ then
18             $K = K + 1$
19             $m_{t,K} = \{\}$ ; $n_{t,K} = \{\}$ ; $n^w_{t,K}\{\}$
20         $m_{t,z} = m_{t,z} + 1$; $n_{t,z} = n_{t,z} + N_d$
21         for each word $w \in d$ do
22             $n^w_{t,z} = n^w_{t,z} + N^w_d$
23 Delete feature tuples of the previous time point.
```

Algorithm 1 shows the procedure of the algorithm at time t. As an additional note, at time point 0, a new cluster is created for the first document and the document is assigned to the newly created topical feature tuple. Afterward, each arriving document in the stream at time t is assigned to either an existing topic or a new topic. The corresponding probability for choosing either an existing topic or a new topic is calculated using (7) and (8). The topic with the highest probability is selected as the label of the current document. Lines 7-10 reflect the add property of feature tuples and lines 13-15 reflect the delete property, where N_d represents the number of words contained in document d and N^w_d represents the frequency with which each word w appears in document d. In the iteration process, information of the current document must first be removed from the topical feature tuple. After all the documents at the current time point have been assigned to the most suitable topic, the iteration ends, and the streaming texts of the next time point are processed. Meanwhile, the feature tuples of the previous moment are deleted.

4 Experiments

In this section, we evaluate the performance of our proposed models by comparison with state-of-the-art models.

4.1 Datasets

We choose two public datasets and their variants for experiments.

- **News**: This dataset comes from the GoogleNews dataset used in GSDMM [29]. News contains 11109 news titles belonging to 152 topics, with an average length of 6.23.
- **Tweets**: This dataset comes from the public dataset Tweets. Tweets includes 30,322 tweets that are closely related to the 269 query items in TREC 2011-2015 microblog tracking. The average length of tweets in the dataset is 7.97.
- **News-T** and **Tweets-T**: In reality, topics change and appear only for a while, so these two datasets resort News and Tweets according to topic and divide them into 16 time points.

These datasets have been preprocessed by word segmentation, stop word removal, lowercase conversion, etc., and the average length matches the size of the short text. Furthermore, after dividing by time points, these static datasets are suitable for streaming text clustering.

4.2 Evaluation metrics

We employ four widely used metrics to evaluate the clustering performance: normalized mutual information (NMI), homogeneity, purity and accuracy [30–32]. In addition, we adopt completeness to measure the performance of the proposed algorithm in the parameter adjustment experiment.

NMI measures the amount of statistical information shared by random variables. These random variables represent the cluster assignment and the basic truth group of documents. NMI is formally defined as follows:

$$NMI = \frac{\sum_c \sum_k n_{c,k} \log(\frac{N \cdot n_{c,k}}{n_c \cdot n_k})}{\sqrt{\sum_c n_c \log(\frac{n_c}{N}) \sum_k n_k \log(\frac{n_k}{N})}} \quad (9)$$

where n_c is the number of documents in class c, n_k is the number of documents in cluster k, $n_{c,k}$ is the number of documents in class c as well as in cluster k, and N is the number of documents in the dataset.

Purity calculate the proportion of the number of correct clustering samples to the total number of samples.

$$Purity = \frac{1}{N} \sum_k \max_c |n_c \cap n_k| \quad (10)$$

Accuracy is used to compare the clustering results with the real classes of the data. Accuracy measures the percentage of assigned correct documents to all clusters.

$$Accuracy = \frac{1}{N} \sum_i^N \delta(c_i, map(k_i)) \quad (11)$$

where k_i and c_i represent the clustering result and the real label corresponding to data x_i respectively, $map(k_i)$ denotes the optimal class label distribution, and the Hungarian algorithm [33] is used to achieve the optimal mapping. In addition, $\delta(a, b)$ is the indicator function. If $a = b$, the value is 1, otherwise it is 0.

Homogeneity represents the proportion of members in a cluster obtained by the algorithm from the same class in the truth value group.

$$Homogeneity = 1 - \frac{H(C|K)}{H(C)} \quad (12)$$

where $H(C|K)$ is the conditional entropy of the class assigned to a given cluster, and $H(C)$ is the class entropy [34].

Completeness is an index of the proportion of members of the same class in the truth group that are divided into the same cluster.

$$Completeness = 1 - \frac{H(K|C)}{H(K)} \quad (13)$$

where $H(K|C)$ is the conditional entropy of the cluster assigned to a given class, and $H(K)$ is the cluster entropy.

The value range of the above metrics is [0, 1], and the higher the score is, the better the clustering performance.

4.3 Methods for comparison

We compare DCSS with the following state-of-the-art models in document clustering.

DTM Dynamic topic model [17] is an extension of LDA that can be used to analyse evolving topics in a document stream. We set $\alpha = 0.01$ for DTM.

Sumblr Sumblr [12] is an online stream clustering algorithm for tweets. With only one pass, it enables the model to cluster tweets efficiently while maintaining cluster statistics.We set $\beta = 0.02$ for sumblr.

oBTM oBTM [25] uses BTM to train documents in each time slice and updates the parameters according to the time point. We set $\alpha = 50/K$, $\beta = 0.01$, and $\lambda = 1$ for oBTM.

DCT DCT [4] enables tracking of the time-varying distributions of topics over documents and words over topics. We initialize α and β to 1 and 0.1, respectively.

MStream MStream [5] clusters text streams by time point with forgetting rules. Only texts within a limited time range are stored in memory. We set $\alpha = 0.03$ and $\beta = 0.03$, and set the maximum storage batch to 2 batches, and the number of iterations to 10.

CTFWPO CTFWPO [35] performs initial clustering assignments based on frequent word pairs in texts, then removes outliers from clusters and reassigns them to more appropriate clusters using semantic similarity. Since the probability calculation in this method is based on MStream, the parameters are set to the same values as those used in MStream.

DTM, Sumblr, oBTM and DCT require a fixed number of topics as input,so we set $K = 300$ and $K = 170$ for the Tweets-T and News-T datasets, respectively. The smaller α is, the more likely the text is to be assigned to a topic with more documents. The larger β is, the more likely the text is to be assigned to a topic with more similar words to itself. To enable the model to generate new topics and to take into account the rarity of words in each short text, the parameters of DCSS are set to $\alpha = 0.2$ and $\beta = 0.04$, and the number of iterations is set to 10.

4.4 Comparison with existing algorithms

In this part, we compare the performance of the proposed model with that of state-of-the-art algorithms. Because the resorted Tweets-T and News-T datasets are more representative of real-life text streams, we compare the clustering results of DCSS and other baselines on these two datasets. Table 2 presents the overall results.

Table 2 Performance of different Methods

Datasets		News-T	Tweets-T	Datasets		News-T	Tweets-T
NMI	DCSS	**0.907**	**0.936**	Purity	DCSS	**0.903**	**0.944**
	MStream	0.881	0.890		MStream	0.821	0.903
	oBTM	0.808	0.800		oBTM	0.753	0.735
	DCT	0.744	0.669		DTM	0.749	0.744
	CTFWPO	0.862	0.892		Sumblr	0.580	0.721
	DTM	0.808	0.803				
	Sumblr	0.723	0.699				
Homogeneity	DCSS	**0.942**	**0.966**	Accuracy	DCSS	**0.764**	**0.809**
	MStream	0.894	0.889		MStream	0.718	0.702
	oBTM	0.833	0.821		oBTM	0.612	0.587
	DTM	0.837	0.824		DTM	0.294	0.201
	Sumblr	0.747	0.893		Sumblr	0.653	0.571

Bold entries signify the superior clustering performance of DCSs

As shown in Table 2, DCSS outperforms all the baselines on both datasets in terms of all measures. Methods (MStream, CTFWP and DCSS) that can infer the number of topics outperform those that require the number of topics to be specified beforehand, which verifies the importance of inferring the number of clusters in short text stream clustering. Compared with CTFWP and MStream, DCSS makes full use of the correlation of text at neighbouring time points, which can effectively alleviate the sparsity of the short text and increase the probability of the text being assigned to the correct topic.

To verify the rationality of combining temporal dependency with the Dirichlet process, we compare the performance of DCSS on the original and resorted datasets in detail. Table 3 shows the average value and deviation of all measurement indicators of DCSS on the four datasets. From Table 3, we can see that DCSS performs much better on both resorted datasets than on the original datasets.

In the resorted datasets, texts belonging to the same topic appear continuously for a certain period of time. When the next hot topic arrives, these texts appear less frequently, but the old topic appears periodically for a period of time with the relevant discussions. However, the overall trend is that topics are constantly changing, and generally, an older topic will eventually disappear over time. Therefore, in News-T and Tweets-T, documents at neighbouring time points have a strong correlation, which is also true in real life. Therefore, DCSS makes reasonable use of the temporal correlation of streaming texts by referring to the clustering result of the last time point to obtain more statistical information and alleviate the data sparsity of the streaming short documents.

In conclusion, DCSS can address the sparsity problem and obtain better clustering results by accounting for the temporal dependency of short text streams.

4.5 Influence of the Number of Iterations

We conducted experiments with the number of iterations, and set the number of iterations varying from 0 to 15 and the other parameters unchanged and selected the NMI and the inferred number of topics as a reference.

Figure 2a shows the NMI value of the DCSS clustering results of the two datasets with different numbers of iterations. When the number of iterations changes from 0 to 1, the NMI tends to be stable. Figure 2b shows the number of clusters of the two datasets inferred under different iteration numbers. As observed in the figure, when the number of iterations is greater than 4, the number of clusters tends to remain unchanged. Streaming texts at a time point can be processed multiple times, which allows the text with similar content to be effectively grouped into the same topic. As the number of iterations increases further, the topic distribution tends to be stable, and the number of topics does not change.

Table 3 Performance of DCSS on original/resorted datasets

	News	News-T	Tweets	Tweets-T
NMI	0.639±0.003	**0.907±0.004**	0.811±0.001	**0.936±0.005**
Homogeneity	0.769±0.005	**0.942±0.003**	0.884±0.003	**0.966±0.005**
Purity	0.602±0.007	**0.903±0.005**	0.796±0.008	**0.944±0.011**
Accuracy	0.100±0.001	**0.764±0.013**	0.462±0.008	**0.809±0.023**

Bold entries signify the superior clustering performance of DCSs

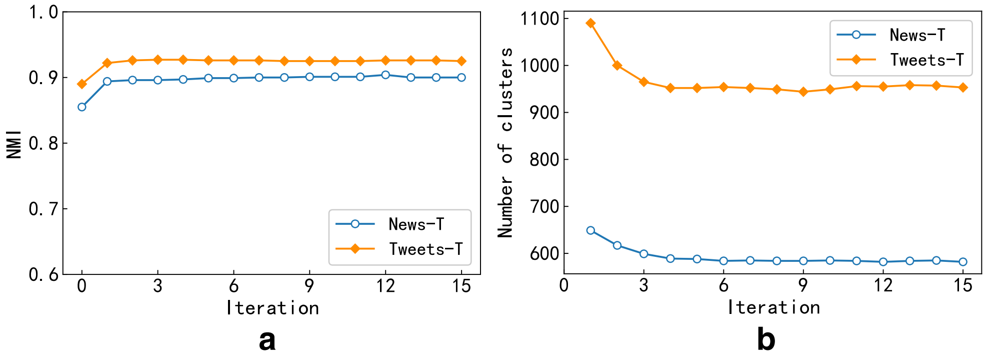

Fig. 2 Influence of the Number of Iterations. **a** NMI. **b** The number of Clusters

4.6 Influence of alpha

In this subsection, we explore the influence of α for DCSS. While fixing the other parameters, we set α to vary from 0.1 to 1.0. We select NMI, homogeneity, completeness, and the inferred number of topics for reference.

Figure 3a shows the change in the NMI under varying α : the NMI is relatively stable. According to with the homogeneity and completeness indicators of the Tweets-T dataset in Fig. 3d, the algorithm in this paper is relatively stable under different α values. Figures 3b and 3c show that as α increases, the number of clusters obtained by clustering also increases. Moreover, Fig. 3c indicates that when α is sufficiently small, the inferred number of topics approaches the true value. According to the CRP [28], this result is reasonable. The hyperparameter α can be understood as the number of people at a virtual table, and this virtual table is equivalent to the situation in which a newly added table may be present for a certain period of time. As α gradually increases, the number of people at this virtual table also increases. The greater the number of people, the more likely newcomers are to choose this virtual table. Therefore, when processing the text stream at each point in time, each text is more likely to be assigned to a new topic as α increases, which leads to more topics inferred by clustering.

4.7 Influence of beta

In this subsection, we explore the influence of β for DCSS. We vary β from 0.01 to 0.20 while fixing the other parameters. The NMI, homogeneity, completeness, and inferred number of topics are selected as references.

Figure 4a shows the change in NMI with different β values. When β is greater than 0.02, the clustering effect of

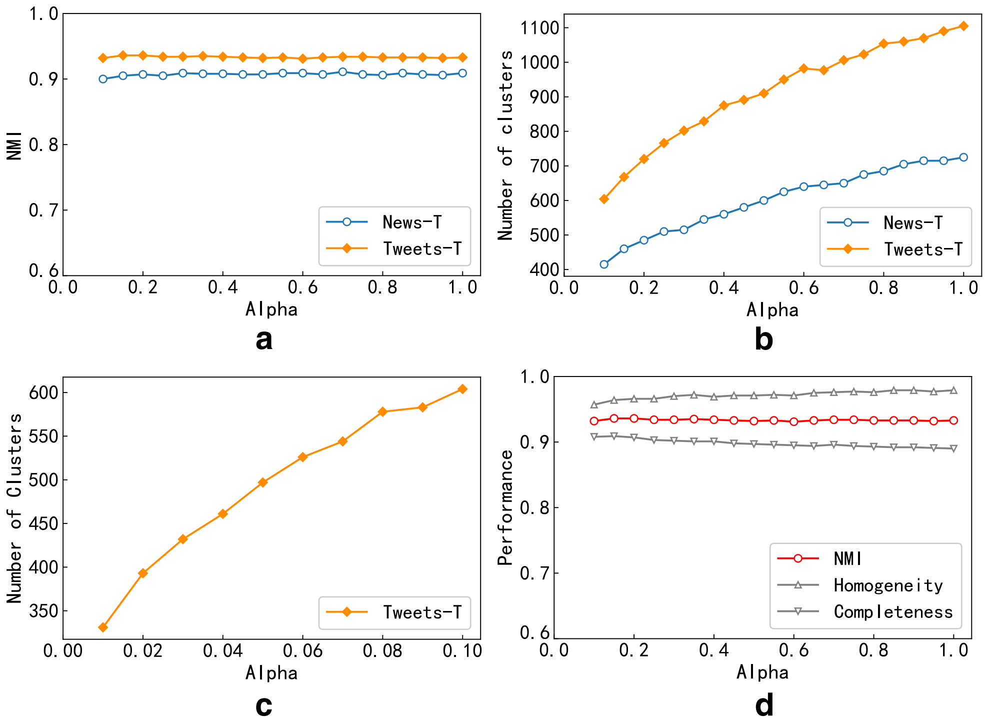

Fig. 3 Influence of Alpha. **a** NMI. **b** The number of Clusters. **c** The number of Clusters of Tweets-T. **d** Performance of DCSS with different values of on α Tweets-T

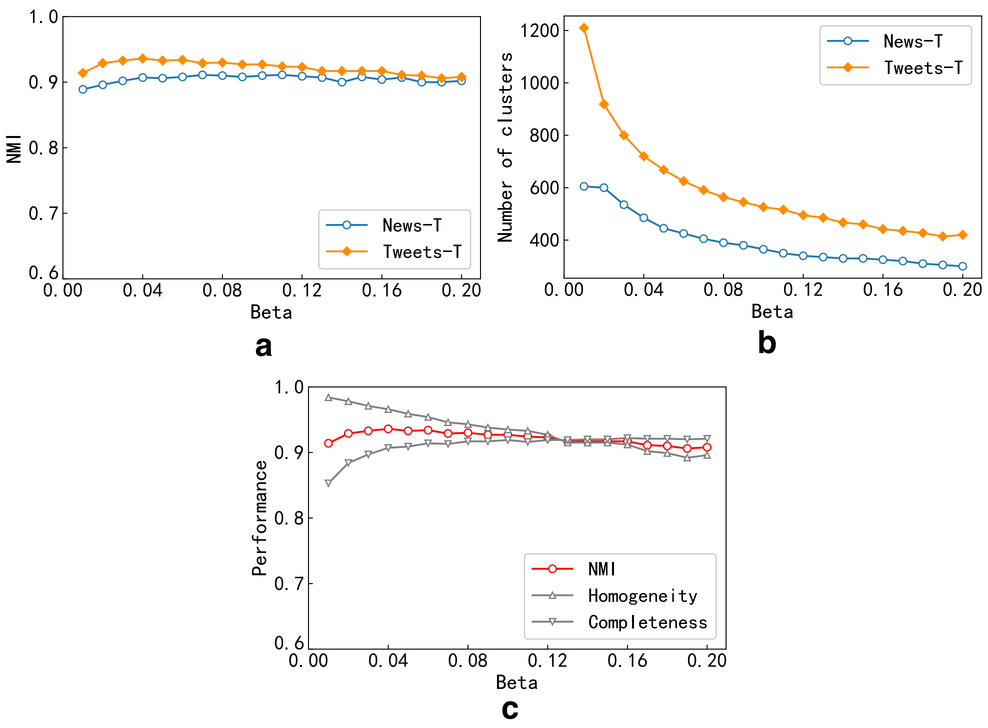

Fig. 4 Influence of Beta. **a** NMI. b The number of Clusters. **c** Performance of DCSS with different values of a α on Tweets-T

DCSS on both datasets is relatively stable. Figure 4c shows that the homogeneity and completeness of the Tweets-T dataset fluctuate under different β conditions, but the values are not excessive from a global perspective.

Figure 4b indicates that the number of clusters inferred by clustering decreases as β increases; this can also be explained by the CRP. β can be regarded as the number of dishes that new customers may be interested in on each table. When β is relatively small, new customers will choose the table to sit down at according to the number of dishes. However, when β is relatively large, even if a customer likes only one dish on the table, he may choose to sit down, which makes people tend to choose a table with more dishes and a greater number of each dish, so the total number of tables will be relatively small. Therefore, when processing the text stream at each time point, each text is more likely to be assigned to an existing topic as β increases, which leads to fewer clusters.

5 Conclusion

This paper proposes a dynamic clustering method for short text streams based on the Dirichlet process (DCSS) that can cope with the sparsity problem of short text and solve the problems of dynamics and topic drift of text streams. DCSS dynamically assigns a batch of arriving documents to existing clusters or generates a new cluster based on the Dirichlet process. More importantly, DCSS incorporates semantic information of the temporal dependence of the streaming texts into the proposed graphical representation model to alleviate the sparsity problem in short text clustering. The experimental results on the resorted datasets prove that the past topic distribution can be used as a prior of the topic distribution of the current time to cope with the sparsity of short texts. We compare the clustering performance with state-of-the-art baselines on public datasets, and verify that DCSS achieves better performance. In the future, we will incorporate a pre-trained language model to improve the performance on short text clustering.

Acknowledgements This research is partially supported by the National Natural Science Foundation of China under grants 62076217 and 61703362.

References

1. Kumar J, Shao J, Uddin S, Ali W (2020) An online semantic-enhanced dirichlet model for short text stream clustering. In: Proceedings of the 58th annual meeting of the association for computational linguistics, pp 766–776
2. Nimala K, Jebakumar R (2019) A robust user sentiment biterm topic mixture model based on user aggregation strategy to avoid data sparsity for short text. J Med Syst 43(4):93
3. Almeida PRL, Oliveira LS, Britto Jr AS, Sabourin R (2018) Adapting dynamic classifier selection for concept drift. Expert Syst Appl 104:67–85
4. Liang S, Yilmaz E, Kanoulas E (2016) Dynamic clustering of streaming short documents. In: Proceedings of the 22nd ACM

SIGKDD international conference on knowledge discovery and data mining, pp 995–1004

5. Yin J, Chao D, Liu Z, Zhang W, Yu X, Wang J (2018) Model-based clustering of short text streams. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pp 2634–2642
6. Teh YW (2010) Dirichlet process.
7. Aggarwal CC, Philip SY, Han J, Wang J (2003) A framework for clustering evolving data streams. In: Proceedings 2003 VLDB conference, pp 81–92. Elsevier
8. Cao F, Estert M, Qian W, Zhou A (2006) 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
9. Yoo S, Huang H, Kasiviswanathan SP (2016) Streaming spectral clustering. In: 2016 IEEE 32nd international conference on data engineering (ICDE), pp 637–648. IEEE
10. Zhong S (2005) Efficient streaming text clustering. Neural Netw 18(5-6):790–798
11. Aggarwal CC, Philip SY (2010) On clustering massive text and categorical data streams. Knowledge and information systems 24(2):171–196
12. Shou L, Wang Z, Chen K, Chen G (2013) 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
13. Kalogeratos A, Zagorisios P, Likas A (2016) Improving text stream clustering using term burstiness and co-burstiness. In: Proceedings of the 9th hellenic conference on artificial intelligence, pp 1–9
14. Terenin A, Simpson D, Draper D (2020) Asynchronous gibbs sampling. In: International Conference on Artificial Intelligence and Statistics, pp 144–154
15. Chu D, Reyers M, Thomson J, Wu LY (2020) Route identification in the national football league: An application of model-based curve clustering using the em algorithm. Journal of Quantitative Analysis in Sports 16(2):121–132
16. Blei DM, Ng AY, Jordan MI (2003) Latent dirichlet allocation. Journal of machine Learning research 3(Jan):993–1022
17. Blei DM, Lafferty JD (2006) Dynamic topic models. In: Proceedings of the 23rd international conference on Machine learning, pp 113–120
18. Wang X, McCallum A (2006) Topics over time: a non-markov continuous-time model of topical trends. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 424–433
19. Wei X, Sun J, Wang X (2007) Dynamic mixture models for multiple time-series. In: Ijcai, vol 7, pp 2909–2914
20. Hoffman M, Bach FR, Blei DM (2010) Online learning for latent dirichlet allocation. In: advances in neural information processing systems, pp 856–864
21. Iwata T, Watanabe S, Yamada T, Ueda N (2009) Topic tracking model for analyzing consumer purchase behavior. In: Twenty-First international joint conference on artificial intelligence
22. Amoualian H, Clausel M, Gaussier E, Amini M-R (2016) 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
23. Zhao Y, Liang S, Ren Z, Ma J, Yilmaz E, deRijke M (2016) 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
24. Yin J, Wang J (2016) A text clustering algorithm using an online clustering scheme for initialization. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp 1995–2004
25. Cheng X, Yan X, Lan Y, Guo J (2014) Btm: Topic modeling over short texts. IEEE Trans Knowl Data Eng 26(12):2928–2941
26. Ahmed A, Xing E (2008) Dynamic non-parametric mixture models and the recurrent chinese restaurant process: with applications to evolutionary clustering. In: Proceedings of the 2008 SIAM international conference on data mining, pp 219–230. SIAM
27. Chen J, Gong Z, Liu W (2019) A nonparametric model for online topic discovery with word embeddings. Inf Sci 504:32–47
28. Blei DM, Griffiths TL, Jordan MI (2010) The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. Journal of the ACM (JACM) 57(2):1–30
29. Yin J, Wang J (2014) A dirichlet multinomial mixture model-based approach for short text clustering. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 233–242
30. Abualigah LM, Khader AT, Hanandeh ES (2018) Hybrid clustering analysis using improved krill herd algorithm. Appl Intell 48(11):4047–4071
31. Abualigah LM, Khader AT, Hanandeh ES (2018) A new feature selection method to improve the document clustering using particle swarm optimization algorithm. Journal of Computational Science 25:456–466
32. Abualigah LMQ (2019) Feature selection and enhanced krill herd algorithm for text document clustering. Springer, Berlin
33. Mills-Tettey GA, Stentz A, Dias MB (2007) The dynamic hungarian algorithm for the assignment problem with changing costs. Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-07-27
34. Rosenberg A, Hirschberg J (2007) V-measure: A conditional entropy-based external cluster evaluation measure. In: Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL), pp 410–420
35. Rakib MRH, Zeh N, Milios E (2020) 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

Wanyin Xu is a graduate student in the Department of software engineering, School of Information Engineering, Yangzhou University, China. His main research interest is short text clustering of natural language processing.

Yun Li received the M. Eng. degree in computer science and technology from Hefei University of Technology, China, in 1991, and the Ph.D. degree in control theory and control engineering from Shanghai University, China,in 2005. He is now a professor in the School of Information Engineering, Yangzhou University, China. He has published more than 70 scientific papers. His research interests include data mining and cloud computing.

Jipeng Qiang received his M.S. and Ph.D. degrees both in computer science from Hefei University of Technology. He is an assistant professor in the School of Information Engineering, Yangzhou University, China. His research interests include pattern mining and text mining.