An Online Semantic-Enhanced Graphical Model for Evolving Short Text Stream Clustering

Jay Kumar, Salah Ud Din, Qinli Yang, Rajesh Kumar, and Junming Shao, *Member, IEEE*

***Abstract*—Due to the popularity of social media and online fora, such as Twitter, Reddit, Facebook, and Wechat, short text stream clustering has gained significant attention in recent years. However, most existing short text stream clustering approaches usually work on static data and tend to cause a "term ambiguity" problem due to the sparse word representation. Beyond, they often exploit short text streams in a batch way and are difficult to find evolving topics in term-changing subspaces. In this article, we propose an online semantic-enhanced graphical model for evolving short text stream clustering (OSGM), by exploiting the word-occurrence semantic information and dynamically maintaining evolving active topics in term-changing subspaces in an online way. Compared to the existing approaches, our online model is not only free of determining the optimal batch size but also lends itself to handling large-scale data streams efficiently. It is also able to handle the "term ambiguity" problem without incorporating features from external resources. More importantly, to the best of our knowledge, it is the first work to extract evolving topics in term-changing subspaces automatically in an online way. Extensive experiments demonstrate that the proposed model yields better performance compared to many state-of-the-art algorithms on both synthetic and real-world datasets.**

***Index Terms*—Clustering, graphical model, microclusters, short text stream, topic evolution, topic modeling.**

Manuscript received May 5, 2021; revised August 9, 2021; accepted August 25, 2021. This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant ZYGX2019Z014; in part by the National Natural Science Foundation of China under Grant 61976044; in part by the Fok Ying-Tong Education Foundation for Young Teachers in the Higher Education Institutions of China under Grant 161062; and in part by the National Key Research and Development Program under Grant 2016YFB0502300. This article was recommended by Associate Editor N. Zhang. *(Corresponding author: Junming Shao.)*

Jay Kumar, Qinli Yang, and Junming Shao are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 313001, China (e-mail: jay@std.uestc.edu.cn; qinli.yang@uestc.edu.cn; junmshao@uestc.edu.cn).

Salah Ud Din is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, also with the Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 313001, China, and also with the Department of Computer, COMSATS University Islamabad, Islamabad 45550, Pakistan (e-mail: salahuddin@std.uestc.edu.cn).

Rajesh Kumar is with the Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 313001, China.

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TCYB.2021.3108897.

Digital Object Identifier 10.1109/TCYB.2021.3108897

I. Introduction

A SHORT text stream is the continuous arrival of short-length documents over time. Nowadays, more and more short text data have been generated on social media and other platforms every day, for example, Twitter, Facebook, QA blogs, and News blogs. Clustering such continuous short text documents has gained increasing attention in recent years due to its diverse applications, that is, news recommendation [1], topic tracing [2], hot topic detection [3], etc. However, due to the unique properties of short text streams such as infinite length, sparse word representation, and topic evolution, clustering short text streams is still a big challenge. During the past decade, many approaches have been introduced to deal with different problems of short text stream clustering.

Challenge 1 (Evolving Number of Topics on Text Streams): For handling text streams with infinite length, initially, static text clustering approaches were transformed to deal with streaming data by considering temporal dependency to discover latent topics [4]–[6]. By transforming the traditional text clustering algorithms, similarity-based approaches were introduced, such as HPStream [7], Feature Weighting *K*-Means [8], spherical *K*-means (SPKMs) [9], COBWEB [10], and ConStream [11]. With a prespecified threshold, similarity-based approaches calculate the similarity scores between the arriving document and existing clusters. It was not long after that, statistical model-based approaches were introduced to deal with text streams, such as the dynamic topic model (DTM) [12], temporal text mining (TTM) [13], temporal Dirichlet process mixture model (TDPM) [14], trend analysis model (TAM) [15], temporal latent Dirichlet allocation (TM-LDA) [16], DPMFP [17], and Gibbs sampling for Dirichlet mixture model (GSDMM) [18], to mention a few. These approaches often require a prespecified number of latent topics, and cannot deal with evolving unknown number of topics in text streams. However, due to the dynamic velocity of streams, determining the evolving number of topics over streams is a nontrivial task.

Challenge 2 ("Term Ambiguity" of Short-Text Word Representation): In addition, unlike long text documents, short text clustering further suffers from the lack of supportive term occurrence to capture semantics [19] due to the sparse word representation. For most existing short text clustering algorithms, such as Sumblr [20], dynamic cluster topic model (DCT) [21], MStreamF [22], and SIF-AE [23], exploiting independent word representation in their cluster models tends

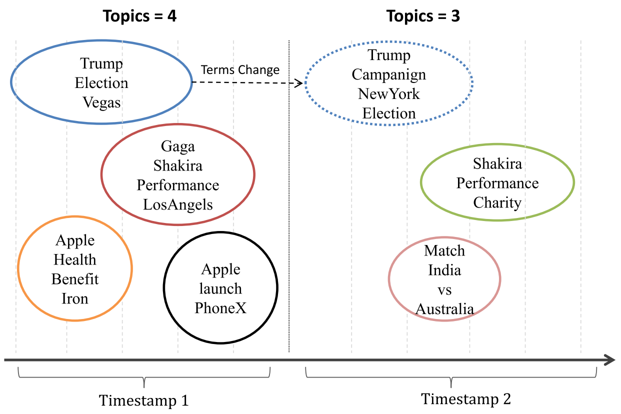


Fig. 1. Illustration of challenges of short text stream clustering with (a) evolving number of topics, (b) term ambiguity, and (c) evolving term distribution problems. Here, the size of the topic represents its active life-span in terms of unit time. For evolving topics, the number of topics is unknown and changes over time. Whereas a topic is represented by its core terms (number of core terms varies from topic to topic), and when a concept drift occurs in the topic's subspace, the distribution of core terms changes.

to cause "term ambiguity" [19] (i.e., the same word has different meanings in different contexts). To deal with this problem, previous works often enrich short text representations by incorporating features from external resources [24], [25].

Challenge 3 (Online Evolving Topic Modeling): Another potential problem of most existing approaches is batchwise processing. The batchwise processing assumes that the instances are interchangeable within a batch[1] [3], [26]. Determining an optimal batch size is also a nontrivial task for different text streams. These algorithms also need to process each batch multiple times to infer the clusters. For online processing, NPMM [27] is proposed to discover evolving number of topics over time and allows finding core terms with the help of external word-embeddings. However, due to the context of terms changes over time, pretrained word embeddings are thus not efficient in real-time streams [28].

To clearly describe the mentioned problems, Fig. 1 shows an example, where the three challenges are depicted. The first problem is the number of topics at timestamp 1 varies from Timestamp 2. The second problem is the "term ambiguity" issue, where "Apple" is a context depending term, enriched in different topics. The third challenging problem is the change in term distribution over time for the topic "Trump Election Campaign."

To tackle the aforementioned issues, we propose an online semantic-enhanced graphical model (OSGM) for evolving short text stream clustering. Compared to the state-of-the-art algorithms, our proposed model can: 1) automatically detect the number of topics over time using the Polya Urn scheme; 2) capture semantics by embedding the evolving term co-occurrence matrix; 3) find the topic-related term-subspace in an online way; and 4) track cluster term evolution using feature-level triangular time decay. The main contributions of this article are highlighted as follows.

1) *Evolving Topic Modeling With Online Term Distribution Exploring:* Instead of batchwise processing and eliminating the constraint of external embeddings, OSGM works in an online way to automatically detect the number of clusters and exploit cluster-document population (see Section IV-B3) to identify the evolving topics in changing term-subspaces.
2) *Semantics Enrichment:* To deal with the "term ambiguity" issue in short text clustering, an evolving term co-occurrence matrix is introduced to capture the relationship between terms for improving the cluster purity. We further exploit the inverse cluster frequency for singular terms as semantic smoothing.
3) *Robustness:* OSGM allows merging highly similar clusters, which yields the number of topics near to actual ones automatically. Moreover, unlike traditional Dirichlet-based nonparametric models, OSGM is more robust to the concentration parameter (Dirichlet process (DP) parameter, cf. Section III-B) and background term noise parameter (Multinomial distribution coefficient, cf. Section III-B).

II. Related Work

A. Model-Based Clustering

Latent Dirichlet allocation (LDA) [29] is the most popular approach introduced to model the generation of topics from the static corpus. LDA is able to extract k number of topics by identifying the subspace of terms from the ocean of term features through constructing a document-topic matrix. However, it cannot deal with the temporal data for text streams. That is, the reason many variants of LDA are proposed for temporal data, such as DTM [12], the dynamic mixture model (DMM) [30], online clustering of text streams (OCTSs) [31], temporal LDA (T-LDA) [16], and streaming LDA (S-LDA) [32]. These models assume that each document contains rich content and, thus, they are not suitable to deal with the short text streams. Later, the Dirichlet multinomial mixture model-based DCT is proposed to deal with short text streams by assigning each document with a single topic [21]. However, these models need a high value of k (number of clusters) to generate clusters through DP. Later on, GSDMM [18] extends the DMM model to infer the number of clusters by integrating collapsed Gibbs sampling. However, most of these models do not explore the evolving topics (clusters) in text streams, where the number of topics usually evolves over time [33], [34].

To automatically detect the number of clusters, Ahmed and Xing [14] proposed a temporal DP mixture model (TDPM). It divides the text stream into many chunks (batches) and assumes that the documents inside each batch are interchangeable. Like other models, it follows the fine-grain clustering model to generate many small clusters. To infer the number of clusters in each batch, GSDPMM [36] is proposed with collapsed Gibbs sampling. In contrast to other models, GSDPMM not only converges faster than LDA but also dynamically assigns the number of clusters over time. However, both TDPM and GSDPMM models do not

[1] A batch is a chunk of instances, objects or documents.

examine the evolving topics by maintaining the active topics. Thereafter, MStreamF [22] is thus proposed by incorporating a forgetting mechanism to cope with cluster evolution and allows processing each batch only one time. However, estimating an optimal batch size is a nontrivial task. Along with the disadvantage of fixed batch size, it also cannot deal with the term ambiguity problem. Recently, the NPMM model [25] is introduced by using the word embeddings to eliminate a cluster generating parameter of the model. Word embedding is useful to remove term ambiguity; therefore, it can represent the relationship between different terms. However, word embeddings of a particular language model need to be pretrained; hence, cannot capture evolving semantics for generating quality clusters. Therefore, capturing semantics by removing term ambiguity is a nontrivial task while dealing with OCTS.

In addition, Zhang *et al.* argue that a text document is often full of general (topic-independent) words and short of core (topic-specific) words. For this purpose, Liu *et al.* [31], [37] proposed term smoothing to reduce the dimensions in model-based approaches, which helps to highlight core terms. NPMM [25] maintains a bit vector to represent the terms of the topics. The bits in the vector change on arriving documents by exploiting word embeddings. However, most previous model-based approaches do not apply feature reduction on the cluster feature (CF) set, instead, each cluster is represented by terms of its documents. Due to the multinomial assumption of DP, these approaches cannot track evolving topics for online text streams.

B. Similarity-Based Clustering

Initially, a general framework to deal with streaming data is proposed by Aggarwal *et al.* [38]. The framework is composed of online and offline clustering modules. It is supposed to assign a much higher number of microclusters than the number of actual macro-clusters. Later on, many popular clustering algorithms are mapped to this general framework. Zhong [9] applied an adaptive SPKM clustering on streaming data. The proposed approach creates unit-length vectors for all the microcluster centroid. However, scalability and sparsity issues still exist [27]. Later on, OSKM [9] is proposed by combining adaptiveness and scalable algorithm on stream text data together. An additional decay mechanism is also designed to analyze the impact on the clustering process. The HPStream [7] clustering algorithm is designed by adopting projected clustering in stream data to tackle the sparsity issue. On the other side, FW-KMeans [8] is also embedded with a generic framework that can assign feature weights to reduce the feature space. Another challenge to exploit the feature weights is to utilize the module for text data. Previous algorithms could not handle the high-dimensional vector space, and the text data contains thousands of unique terms in the active stream. A factor is introduced to adjust the term distribution in the cluster. The popular algorithm ConSTREAM [11] is exploited to deal with text data in the stream. A *cluster droplet* is introduced as CF, which represents only those terms related to local space. By using the decay mechanism, outliers are also detected by calculating the cosine similarity between microclusters. In general, the limitation of similarity-based clustering approaches is that they require a manually fixed threshold for assigning a document to an old or new cluster. Moreover, sparsity is another issue for short-text clustering when adopting threshold-based approaches.

III. Preliminaries

A. Notations and Notions

For the continuous-time representation, we denote time as $T = t_1, t_2, \ldots, t_\infty$. A text stream S_t is a sequence of arriving documents over time, $S_t = \{d_t\}_{t=1}^{\infty}$. Due to the unknown length of the stream, we represent time up to infinity. Here, an arrived document d at time t is represented as d_t. Each document in the stream contains N words denoted as: $d_t = \{w_1, w_2, \ldots, w_N\}$. Generally, every document d_t has different length; therefore, N varies from document to document. The main objective of the text clustering task is to group the similar documents into a common cluster. Formally, we can represent the clusters as: $Z = \{z_t\}_{t=1}^{\infty}$. Like the length of a given stream, we do not know the number of topics and, thus, we represent the number of clusters up to infinity. Here, each cluster contains documents, which is denoted as $z_t = \{d_1^{z_t}, d_2^{z_t}, \ldots, d_m^{z_t}\}$. Specifically, for short text clustering, a document only belongs to a single topic; therefore, $z_i \cap z_j = \phi$, where $i \neq j$.

B. Dirichlet Process

The DP is a nonparametric type of stochastic process, used to model the random generation of a distribution represented as $G_0 \sim \mathrm{DP}(\alpha, \mathcal{G})$. Here, $\mathcal{G}$ is the base distribution, and G_0 is the drawn sample from the distribution. Interestingly, the drawn sample itself is a distribution; thus, also referred to as distribution over distribution. The drawing concentration is controlled by α parameter.

Poly Urn Scheme: It is the procedure to draw a sequence of samples $G_1, G_2, \ldots$ from a distribution [39]. The scheme is summarized as follows:

$$G_n|G_{1:n-1} \sim \frac{\alpha}{\alpha+n-1} + \frac{\sum_{k=1}^{n-1}\delta(G_n - G_k)}{\alpha+n-1}. \qquad (1)$$

Here, $\delta(x) = 1$ if $x = 0$, and $\delta(x) = 0$ otherwise. Initially, we assume that there exists an empty urn and predefined distribution of colors. As we draw a color from a base distribution (i.e., $G_1 \sim \mathcal{G}$), we put a ball of drawn color into the empty urn. At this defined point, the urn is no longer empty. In the next turn, we either draw a color that already exists in the urn with a probability of $(n - 1/\alpha + n - 1)$ or draw a new color with probability of $(\alpha G_0/\alpha + n - 1)$. The same color may appear more than once due to the repetitive drawing procedure. This defines that we have n number of draws and K unique colors. The procedure of partitioning n draws into K clusters is defined by a well-known Chinese restaurant process (CRP) [40].

Chinese Restaurant Process: Assume that there exists a restaurant with an infinite number of tables and each table surrounds an infinite number of empty chairs for the customers. Initially, all tables are empty; therefore, the first arriving customer has to sit on the first table. Later on, the nth customer either chooses to sit on any occupied table with a probability of $(n_k/\alpha + n - 1)$ or chooses an empty table with a probability of $(\alpha/\alpha + n - 1)$. Here, n_k is the number of customers sitting on a specific table. A new customer tends to be attracted toward a highly crowded table, which is called richer-gets-rich characteristics. The samples of CRP are drawn from the distribution $\mathcal{G}_{\text{crp}}$ and the stick-breaking process shows the property of the distribution, where the procedure ranges toward infinity

$$\mathcal{G}_{\text{crp}}(G) = \sum_{k=1}^{\infty} \theta_k \delta(G - G_k), \quad G_k \sim \mathcal{G}. \tag{2}$$

The mixture weights $\theta = \{\theta_k\}_{k=1}^{\infty}$ can be formalized by $\theta \sim GEM(\gamma)$ [41]. We exploit (2) for the generative process of the DP multinomial mixture model (DPMM) [36] as follows:

$$\begin{aligned} z_d|\theta &\sim \text{Mult}(\theta) \quad d = 1, \ldots, \infty \\ G_k|\beta &\sim \text{Dir}(\beta) \quad k = 1, \ldots, \infty \\ d\big|z_d, \{G_k\}_{k=1}^{\infty} &\sim p\big(d|G_{z_d}\big) \end{aligned}$$

where z_d is the assigned documents to the cluster, which are multinomial distributed. The probability of document d generated by topic z is summarized as

$$p(d|G_z) = \prod_{w \in d} \text{Mult}(w|G_z). \tag{3}$$

Here, the naive Bayes assumption is considered, where words in a document are independently generated by the topic. In other words, the position of words in a document is not considered while calculating the probability.

By following the CRP to design a generative process for infinite number of tables (topics), we combine two probabilities: 1) probability of topic popularity (defined by CRP) and 2) similarity between the document and the topic in terms of word distribution:

$$p(z_d|G_z) = p(G_z).p(d|G_z). \tag{4}$$

Here, $p(G_z)$ defines the probability of topic popularity (in CRP it is table popularity), which is $(n_k/\alpha + n - 1)$ for choosing the existing topic and $(\alpha/\alpha + n - 1)$ for choosing a new topic. The term $p(d|G_z)$ represents the similarity between topic (table) and document (customer), multinomial distribution with DP is used for existing topic and for the new topic as well.

IV. Proposed Approach

In this section, we give a detailed description of the CF set and then introduce our OSGM for evolving short text stream clustering.

A. Cluster Feature Set

A microcluster in the stream is represented by the CF set, which contains different statistical values about the instances within a microcluster. Most similarity-based and model-based methods follow the vector space model (VSM) to represent the CF space [42]. However, a topic needs to be represented in the term-changing subspaces. Here, an extended CF is employed, where each cluster is represented by words of related documents. In our model, a CF set is defined as a 7-tuple $\{m_z, n_z^w, cw_z, n_z, l_z, u_z, \text{ta}_z\}$. The description of each tuple is given in Table I. The CF has the important addable property which allows a cluster to update incrementally over time.

TABLE I
Symbols and Notations

Symbol	Definition
D	total number of active documents
V_d	unique terms / vocabulary in document d
V_z	unique terms / vocabulary of cluster z
$\mathcal{L}$	average length of document in stream
$\bar{V_z}$	average vocabulary size of active clusters $(V_z \cap V_d \neq \{\})$
cw_{ij}	co-occurrence of words w_i and w_j
N_d^w	occurrence of word w in document d
N_d	total number of words in document d
n_z^w	term frequency of w in cluster z
cw_z	word to word co-occurrence matrix
z_d	assigned cluster of document d
m_z	number of documents in cluster z
n_z	number of words in cluster z, $\sum_{w \in z} n_z^w$
l_z	decay weight of cluster z
u_z	last updated time stamp of cluster z
ta_z	arriving timestamps of the words in clusters z
$V_{z \cup d}$	vocabulary size of $V_z \cup V_d$

Definition 1: A document d can be added to a cluster z by using the *addable property* to update the cluster

$$\begin{aligned} m_z &= m_z + 1 \\ n_z^w &= n_z^w + N_d^w \quad \forall w \in d \\ cw_z &= cw_z \cup cw_d \\ n_z &= n_z + N_d \\ \text{ta}_z &= \text{ta}_z \uplus \text{ta}_d \quad \forall w \in d. \end{aligned}$$

Here, cw_d is the co-occurrence of each word with every word of the same document, and N_d represents the number of total words in the document. $\uplus\, \text{ta}_d$ merges the arrival time of each document term (the example is provided in the supplementary material). The complexity of updating a cluster by adding a document is $O(\mathcal{L}^2)$, where $\mathcal{L}$ is the average length of the documents. This property is useful to update the evolving microcluster in the text stream clustering.

B. Online Semantic-Enhanced Graphical Model

The probabilistic graphical model (PGM) is used to represent the stochastic generation of objects (short text document in our case) with Bayesian probabilistic modeling. To model the topic generation for the text stream clustering problem, the most crucial task is to define the relationship between documents and topics (clusters) in terms of a probability distribution. Fig. 2 shows our graphical model: OSGM. We extend and transform the MStreamF model from batchwise

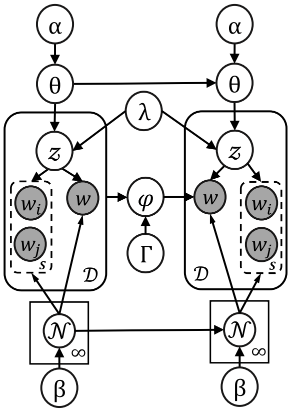


Fig. 2. Graphical model of OSGM. The shaded are observed and unshaded are latent entities. z represent the topic, θ is distribution of topics with α as the concentration parameter. $\mathcal{D}$ is collection of documents with w words. $\mathcal{N}$ represents the multinomial distribution of words over D with β as the distribution parameter. λ is the exponential decay function parameter. Γ represents the threshold value of triangular function φ.

to the online process. In the graphical model, θ represents the distribution of topics z drawn with α as concentration parameter and $\mathcal{N}$ is the distribution of terms over generated topics with β as the term coefficient. φ controls the change of topic-related terms over time with Γ as the term recency threshold. The decay process to remove the outdated topics from the model is controlled by λ as a decay factor. The generative process for topic generation in the stream is presented as follows.

1) Sample $\theta_t \sim Dirichlet(\alpha, \theta_{t-1})$.
2) For each document d:
   a) draw $\mathcal{N}_{t,z}|\beta_{t,z} \sim Dirichlet(\beta_{t-1,z}, \mathcal{N}_{t-1,z})$;
   b) sample $z_d \sim Multinomial(\theta_t)$;
   c) for each word or co-occurrence of word:
      i) $w_d \sim Multinomial(\mathcal{N}_{t,z}|z_d)$.

The characteristics of OSGM are four-fold: 1) eliminate term ambiguity by calculating document-cluster similarity; 2) handle evolving number of topics[2]; 3) determine the evolving topics in changing term-subspaces; and 4) remove outdated topics. The pseudocode of our approach is given in Algorithm 1.

1) Document-Cluster Similarity: Initially, the first arriving document creates its own cluster. For the next arriving document, either it should be added in any active cluster of model ($\mathcal{M}$) or a new cluster is created for it. For calculating the similarity between a document and an existing cluster, we transform (4) and derive the probability of an arriving document d to choose an existing cluster z as

$$p\left(z_d = z|\vec{z}, \vec{d}, \alpha, \beta\right) = \left(\frac{m_z}{D - 1 + \alpha D}\right) \times \left(\frac{\prod_{w\in d}\prod_{j=1}^{N_d^w}\left(n_z^w \cdot lCF_w\right) + \beta + j - 1}{\prod_{i=1}^{N_d} n_z + (\beta V_{z\cup d}) + i - 1}\right) \times \left(1 + \sum_{w_i\in d \wedge w_j\in d} cw_{i,j}\right). \quad (5)$$

Algorithm 1: OSGM

```
Input: $S_t : \{d_t\}_{t=1}^{\infty}$ , $\alpha$ : concentration parameter, $\beta$ :
       pseudo weight of term in cluster, λ : decay factor,
       Γ : feature decay threshold
Output: Cluster assignments $z_d$
initialize $\mathcal{M} = \phi$                 // empty model
$t_c = 0$                                // model timestamp
while $d_t$ in $S_t$                      // start streaming
do
    $t_c = t_c + 1$
    $\mathcal{M} = updateActiveClusters(\lambda, \mathcal{M})$   // Alg. 3
    $\mathcal{M} = updateTermsSubspace(\Gamma, \mathcal{M})$   // Alg. 2
    foreach $z_i \in \mathcal{M}$ do
        if $V_d \cap V_z \neq \phi$ then
            $P_{Z_i} = p(z_d = z_i, d_t)$ using Equation (5)
        end
    end
    $i = \arg\max_i(P_{Z_i})$
    $P_{Z_n} = p(z_d = z_{new}, d_t)$ using Equation (8)
    if $P_{Z_i} < P_{Z_n}$ then
        // create new cluster
        $m_{z_n} = 1$, $n_{z_n}^w = N_{d_t}^w$
        $cw_{z_n} = cw_{d_t}$ , $n_{z_n} = N_{d_t}$
        $l_{z_n} = 1$, $u_{z_n} = t_c$, $ta_{z_n} = ta_{d_t}$
        $\mathcal{M} = \mathcal{M} \cup z_n$
    else
        update $z_i$ using Definition 1
        $l_{z_i} = 1$, $u_{z_i} = t_c$
    end
end
```

The derivations of the equation are provided in the supplementary material. The first term $(m_z/D - 1 + \alpha D)$ represents $p(G_z)$ of (4). Here, D is the number of current documents in active clusters,[3] Whereas, α is the concentration parameter of the model. We designed two probabilities to define $p(d|G_z)$ of (4). The first part captures similarity in singular term space [the middle term of the (5)] and the second part captures similarity in semantic term space (co-occurrence of terms) to solve the term-ambiguity problem. The middle term is based on the multinomial distribution [see (3)] with a pseudo weight of words β defines the homogeneity between a cluster and a document. Previously, the occurrence of a term in cluster n_z^w is used while calculating the homogeneity (similarity). However, in the static environment inverse document frequency is used to calculate the term importance in global space. Here, in the dynamic environment, we define a similar weight measure, called inverse cluster frequency ICF_w, to calculate term importance, defined as

$$ICF(w \in d) = \log\left(\frac{|Z|}{|w\epsilon V_z|}\right). \quad (6)$$

[2] Clusters and topics are interchangeably used in this article.

[3] Active clusters refer to those clusters which are not yet deleted from the model.

The denominator part of (6) is the number of those clusters which contains the word w, and $|Z|$ is total number of active clusters. The occurrence of a term w exists in more clusters, lesser the importance it has. If a term w is found in very few clusters, it signifies its high importance. To deal with the "term ambiguity" issue in short text clustering, an evolving term co-occurrence matrix is introduced to capture the relationship between terms for improving the cluster purity. For example, assume that we have two clusters $C1$ and $C2$

$$n^w_{C1} = \{w_1 : 3, w_2 : 4, w_1 : 3, w_3 : 1, w_4 : 2\}$$
$$n^w_{C2} = \{w_1 : 3, w_2 : 4, w_1 : 3, w_5 : 1, w_6 : 2\}$$
$$cw_{C1} = \left[\begin{array}{ccccc} w_1 & w_2 & w_3 & w_4 & \\ - & 0.5 & 0 & 0 & w_1 \\ 0.5 & - & 0.25 & 0 & w_2 \\ 0 & 0.75 & - & 0.5 & w_3 \\ 0 & 0 & 0.5 & - & w_4 \end{array}\right]$$
$$cw_{C2} = \left[\begin{array}{ccccc} w_1 & w_2 & w_5 & w_6 & \\ - & 0 & 0 & 0.5 & w_1 \\ 0 & - & 0.75 & 0 & w_2 \\ 0 & 0.25 & - & 0.5 & w_5 \\ 0.5 & 0 & 0.5 & - & w_6 \end{array}\right].$$

We assume that $d' = \{w_1 : 1, w_2 : 1, w_7 : 1\}$ as an arriving document and calculate the similarity of d' with both clusters. We can observe that w_1 and w_2 are common between both clusters. If we do not consider the co-occurrence, then the probabilities of both clusters are the same; however, w_1 and w_2 co-occurred in $C1$. Therefore, the third part $(1 + \sum_{w_i \in d \wedge w_j \in d} cw_{i,j})$ in (5) defines the weight of term co-occurrence between the cluster and a document. Formally, we define a value of an entry $cw_{i,j}$ in the co-occurrence matrix as follows:

$$cw_{i,j} = \frac{\sum\limits_{d' \subseteq z} n^{w_i}_{d'}}{\sum\limits_{d' \subseteq z} n^{w_i}_{d'} + \sum\limits_{d' \subseteq z} n^{w_j}_{d'}} \quad \text{s.t.}\big(w_i, w_j\big) \in d'. \tag{7}$$

Here, $n^{w_i}_{d'}$ is the frequency count of word w_i in the document d'. The ratio between w_i and w_j must satisfy $cw_{i,j} + cw_{j,i} = 1$, where $i \neq j$ (an example is discussed in the supplementary material). We calculate the term co-occurrence weight of those terms which are common in the cluster z and document d. Therefore, if the size of the CF set (discussed in Section IV-A) is $|V_z|$, then it is not necessary that the co-occurrence matrix would be $|V_z| \times |V_z|$.

2) Automatic Cluster Creation: Initially, the first arriving document creates its own cluster. To identify new topics over time, we need to define the probability that can automatically create a new cluster for the new topic if the incoming document in the stream does not relate to any existing active cluster. For automatic cluster creation, we transform (4) and derive the probability as follows:

$$p\Big(z_d = z_{new} | \vec{z}_{\neg d}, \vec{d}, \alpha, \beta\Big) = \left(\frac{\alpha D}{D - 1 + \alpha D}\right) \times \left(\frac{\prod_{w \in d} \prod_{j=1}^{N^w_d} \beta + j - 1}{\prod_{i=1}^{N_d} (\bar{V}_z \beta) + i - 1}\right). \tag{8}$$

Algorithm 2: Update Cluster Term Subspace

```
   /* Update weight of terms in each
      active cluster                              */
1  Function updateTermsSubspace(Γ, $\mathcal{M}$)
2      $S_z = \Delta f(1)$ using Equation (9)
3      foreach $z \in \mathcal{M}$ do
4          $E_z = \Delta f(m_z)$ using Equation (9)
5          $Age_z = E_z - S_z + 1$
6          foreach $(w, ta_w) \in ta_z$ do
               // sum arrival timestamps of w
7              $sumta_w = \sum_{t \in ta_w} t$
8              $recency_w = (sumta_w \times 100)/Age_z$
9              if $recency_w < \Gamma$ then
10                 remove$(w, z)$
11             end
12         end
13     end
14     return $\mathcal{M}$
```

Here, $\bar{V_z}$ is the average vocabulary size of active clusters. αD represents the pseudo number of documents and β helps to calculate pseudo term similarity with a new cluster. If this small probability of an arriving document is greater than the probability of existing clusters, then the model creates a new cluster containing the arrival document (see line 15 of Algorithm 1).

3) Online Evolving Topic Extraction in Changing Term-Subspaces: The previous study [43] argues that a topic-related text document has few core terms to represent a topic. Whereas, for some topics, the distribution of core terms also changes over time. To highlight core terms and term subspace change in an online way, we first maintain the arriving timestamps of each term. We then calculate the age of the cluster by adopting a triangular number of its size (see line 5 of Algorithm 2). Afterward, we compare the arrival timestamps of each term with the age of the cluster to reduce noisy terms. The triangular time is originally employed to fade out outdated microclusters [44], defined as

$$\text{Triangular Number } \Delta f(T) = \Big(\Big(T^2 + T\Big)/2\Big). \tag{9}$$

Here, T is the timestamp number. The recency score of a term is measured by the ratio of the sum of arrivals and the age of the cluster (see line 8 of Algorithm 2). For example, cluster z contains nine documents, then the age of cluster z will be

$$\text{Age}_z = \Delta f(9) - \Delta f(1) = \Big(\Big(9^2 + 9\Big)/2\Big) - \Big(\Big(1^2 + 1\Big)/2\Big).$$

Let us assume $D1$, $D2$, and $D8$ belong to z which are:

D_1: "entry performance of Lady gaga."

D_2: "Lady gaga stole the heart by exploding performance."

D_8: "Gaga and shakira rocked!"

and their cluster timestamps[4] are 1, 2, and 8, respectively. The arrivals of terms (ta) in clusters are stored as

$$\text{ta}_{\text{gaga}} = \{\text{timestamp} : 1, \text{timestamp} : 2, \text{timestamp} : 8\}$$

[4] cluster timestamp is equal to number of documents within respective cluster.

and the sum of time arrivals is $\text{sum}(\text{ta}_{\text{gaga}}) = \sum_{t \in \text{ta}_{\text{gaga}}} t$ To calculate the recency score of *gaga*, which will be

$$\text{recency}_{\text{gaga}} = ((1 + 2 + 8) \times 100)/\text{Age}_z$$

if the recency is less than our defined threshold Γ, then we consider that this term cannot represent the current concept of the cluster. In other words, if a term is frequently observed in the cluster with the passage of a cluster time span, then the term is useful to represent the current concept of terms (see line 9 of Algorithm 2). It can be analyzed that more terms become less important when new documents are added in a cluster. In this way, the parameter β of multinomial distribution heavily affects the probability due to the increasing number of unimportant terms. With the triangular number time, important terms are highlighted and noisy terms are filtered out gradually over time. The entire procedure is given in Algorithm 2.

4) Deleting Outdated Clusters: Followed by clusters creating process, the model should maintain only active clusters (current concept) by deleting outdated clusters (old concept). Usually, in a real-time environment, each topic may have a different active time span. For example, "USA-election" may be active on social media for two months, whereas, tweets on a particular "football match" may be active for two days. Many existing approaches often delete old clusters using some forgetting mechanisms (e.g., decay rate), some deletes instances of old batches [9], [11], [22], [27]. The life span of each document depends on the active time period of a topic, therefore, in contrast to deleting old batches, we adopt the forgetting mechanism cluster importance score based on stream velocity. Specifically, the importance of each cluster is decreased over time if it is not updated. For this purpose, we apply an exponential decay function

$$l_z = l_z \times 2^{-\lambda \times (t_c - u_z)}. \tag{10}$$

Here, l_z represents the importance score of a cluster z and t_c is current timestamp of model. Initially, we set $l_z = 1$ of each new cluster (see line 18 of Algorithm 1). The importance score of a cluster decreases over time if it is not receiving any document. If the score of a cluster approximately reaches 0 (see line 7 of Algorithm 3), then the cluster is processed to be removed from the model, that is, it cannot capture recent topic in the text stream. Algorithm 3 shows the step-by-step procedure.

5) Merging Clusters: As we discussed that β parameter of multinomial distribution is responsible for calculating the homogeneity (similarity). However, due to the exceeding noisy terms over time in a cluster, the probability of noisy terms may become dominant over core terms. This leads the model to create many microclusters for a single topic. Previous approaches iterate the stream multiple times to infer the number of clusters. In contrast, we incorporate the cluster merging process to check if an outdated cluster can be merged with an active cluster by calculating the probability using (5) between two clusters. Algorithm 3 shows the step-by-step procedure to merge an outdated cluster with an active cluster. The two clusters are merged using Definition 1.

Algorithm 3: Update Active Clusters

```
   /* Update weight of all active clusters
      */
1  Function updateActiveClusters($\lambda$, $\mathcal{M}$)
2      $Old = \phi$
3      foreach $z \in \mathcal{M}$ do
4          calculate $l_z$ using Equation (10)
5          if $l_z \approx 0$ then
6              $Old = Old \cup z$
7          end
8      end
9      foreach $z_i \in Old$ do
10         foreach $z_{act} \in (\mathcal{M} - Old)$ do
11             if $V_{z_{act}} \cap V_{z_i} \neq \phi$ then
12                 $P_{Z_i} = p(z_d = z_{act}, z_i)$ using Equation (5)
13             end
14         end
15         $max = \arg\max_i (P_{Z_i})$
16         $P_{Z_n} = p(z_d = z_{new}, z_i)$ using Equation (8)
17         if $P_{Z_{max}} < P_{Z_n}$ then
18             $\mathcal{M} = \mathcal{M} - z_i$
19         else
20             merge($z_i$,$Z_{max}$) using Definition 1
21         end
22     end
23 return $\mathcal{M}$
24 End Function
```

Complexity Analysis: The OSGM algorithm always maintains an average $\bar{K}$ number of current topics. Every CF set stores an average $\bar{V}$ number of words in n_z^w, $2 \times |\bar{V}|$ timestamps of each word in ta_z, and at most $|\bar{V}_z| \times |\bar{V}_z|$ in cw_z. Thus, the space complexity of OSGM is $O(\bar{K}(3\bar{V} + \bar{V}^2) + VD)$, where V is the size of active vocabulary and D is the number of active documents. On the other side, OSGM calculates the probability of arriving document with almost each active cluster (see line 8 of Algorithm 1). In addition, it also eliminates outdated vocabulary by checking each CF. Therefore, the time complexity of OSGM is $O(\bar{K}(\mathcal{L}\bar{V}))$, where $\mathcal{L}$ is the average size of the arriving document.

V. Experiments

A. Datasets

To evaluate the performance of the proposed algorithm, we conduct experiments on two real and two synthetic datasets. These datasets were also used in [21], [22], [25], [36], [45], and [46], to evaluate short text clustering models. In the preprocessing step, we removed stop words and converted all text into lowercase, and stemming.

1) *News (Ns):* This dataset is collected by [47]. It contains 11 109 titles of news distributed over 152 topics.
2) *Tweets (Ts):* This dataset contains 30 322 tweets, which are relevant to 269 topics in the TREC[5] microblog.
3) *News-T (Ns-T):* Naturally, we may find a situation, where topics in social media appear only for a certain

[5] http://trec.nist.gov/data/microblog.html

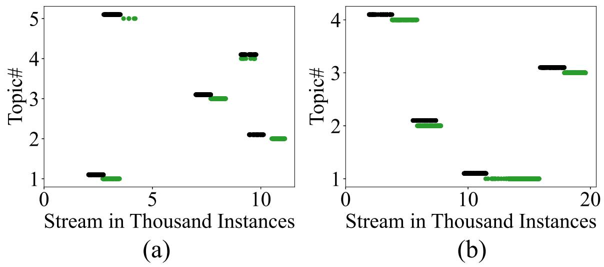


Fig. 3. In News-T and Tweets-T datasets, there are five and four main topics, respectively. Each topic is divided into two subtopics and the plot shows the instances of these subtopics in which change in term distribution occurs. (a) News-T. (b) Tweets-T.

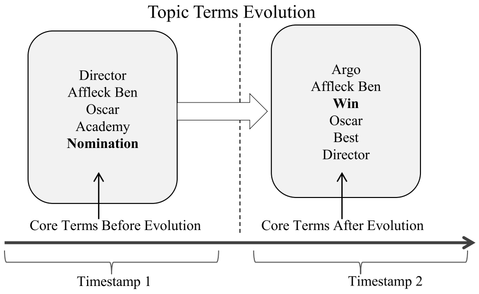


Fig. 4. Changes of core terms for a topic from news.

TABLE II
STATISTICS OF EACH DATASET WHERE DS REPRESENTS THE NAME OF THE DATASET, $|D|$ REPRESENTS THE TOTAL NUMBER OF DOCUMENTS, $|V|$ IS THE NUMBER OF UNIQUE TERMS, $|T|$ IS THE TOTAL NUMBER OF TOPICS, AND $\mathcal{L}$ IS THE AVERAGE DOCUMENT LENGTH

Ds	$\|D\|$	$\|V\|$	$\|T\|$	$\mathcal{L}$
News	11109	8110	152	6.23
Tweets	30322	12301	269	7.97
News-T	11109	8110	147	6.23
Tweets-T	30322	12301	265	7.97

period and then disappear. The next thing to observe is that in many situations, the distribution of topic-related terms changes. We constructed a synthetic dataset by injecting two mentioned conditions in the News dataset as follows.

a) We sorted documents by topics. After sorting, we then divide the dataset into 16 equal chunks and shuffled them.
b) We analyzed five main topics whose term distribution changed in subtopics over time. To depict change in term distribution, we combined subtopics and placed them at a continuous position.

Fig. 3(a) shows the instance position of five topics in the stream. Here, black point instances show the initial distribution of terms of a new topic, whereas, the green points depict instances of the term distributional change (concept drift) of the same topic. An example from this dataset is given in Fig. 4.

4) *Tweets-T (Ts-T):* We repeat the same process for the *Tweets* dataset to construct a synthetic dataset, as shown in Fig. 3(b).

The statistics of the datasets are reported in Table II. Here, both datasets contain less than ten average document lengths ($\mathcal{L}$), which assure datasets fit for short text analysis. Specifically, the News dataset contains a lower average document length, compared to the Tweets dataset, because the dataset was constructed by collecting headlines of news streams.

B. Methods of Comparison

For the deep analysis, we compare two variations of our proposed approach: 1) OSGM and 2) OSGM-ES with baselines. The former approach includes semantic smoothing of inverse clustering frequency, whereas the latter approach excludes it. We have selected five state-of-the-art representative algorithms for text stream clustering to comparison.

1) *DMM* [47] is an initial model based on multinomial Dirichlet modeling to simulate CRP.
2) *OSDM* [48] is the most recent model to deal with an infinite number of topics in an online way.
3) *DTM* [12] is an extension of LDA, which traces the evolution of hidden topics from corpus over time. It was designed to deal with sequential documents.
4) *GSDMM* [47] is a Dirichlet multinomial mixture model for short text clustering, which does not consider the temporal dependency of instances, Whereas, the number of clusters is inferred with the iterative Gibbs sampling procedure.
5) *MStreamF* [22] is a recent model to deal with the infinite number of latent topics in short texts while processing one batch at a time. Two models of MStreamF were proposed, one with the one-pass clustering process, and another with Gibbs sampling. We refer to the former algorithm as MStreamF-O (MF-O) and the latter as MStreamF-G (MF-G).
6) *NPMM* [25] is the latest model to deal with the infinite number of latent topics in short texts while exploiting pretrained Glove word-embeddings to capture semantics. The authors proposed two variations of their algorithm, one with an iterative process referred to as NPMM-G, and another without it which is NPMM-O.

Our algorithm is implemented in Python, and the code is publicly available at https://github.com/JayKumarr/OSGM.

C. Evaluation Measures

We adopted four highly used evaluation metrics for deep analysis of all algorithms, which include Purity (Pur), V-Measure (V-M), Homogeneity (Homo), and normalized mutual information (NMI).

D. Parameter Setting

We try to find the optimal parameter values for best results for most algorithms with grid search. Finally, we obtain the best results on $\alpha = 0.01$ for DTM, $\alpha = 0.3$ and $\beta = 0.3$ for DMM. The DTM and DMM need the fixed number of clusters as input; therefore, we set $K = 300$ and $K = 170$ for Tweets and News, respectively. For MStreamF-O and MStreamF-G, we set $\alpha = 0.03$ and $\beta = 0.03$. By following [22], we set

TABLE III
PERFORMANCE OF DIFFERENT ALGORITHMS ON FOUR DATASETS IN TERMS OF DIFFERENT MEASURES, INCLUDING NMI, V-MEASURE (V-M), HOMOGENEITY (HOMO.), AND CLUSTER PURITY (PUR.)

Alg.	Ds.	Evaluation Measures			
		NMI	V-M	Homo	Pur.
OSGM	Ns	0.815	0.815	0.893	0.824
OSGM-ES		0.822	0.822	0.900	0.841
OSDM		0.814	0.805	**0.950**	**0.907**
MF-G		0.780	0.779	0.751	0.653
MF-O		0.685	0.684	0.654	0.552
DTM		0.800	0.800	0.822	0.749
NPMM-O		0.809	0.809	0.802	0.710
NPMM-G		**0.843**	**0.843**	0.834	0.745
DMM		0.592	0.592	0.595	0.474
GSDMM		0.821	0.821	0.751	0.605
OSGM	Ts	0.822	0.822	0.879	0.819
OSGM-ES		**0.836**	**0.836**	**0.910**	**0.855**
OSDM		0.822	0.831	0.890	0.829
MF-G		0.795	0.793	0.738	0.801
MF-O		0.746	0.744	0.695	0.529
DTM		0.800	0.800	0.822	0.749
NPMM-O		0.769	0.769	0.758	0.634
NPMM-G		0.821	0.821	0.813	0.696
DMM		0.392	0.623	0.549	0.141
GSDMM		0.798	0.798	0.728	0.581
OSGM	Ns-T	**0.854**	**0.854**	0.919	0.878
OSGM-ES		0.853	0.853	**0.958**	**0.936**
OSDM		0.854	0.854	0.901	0.868
MF-G		0.848	0.848	0.829	0.705
MF-O		0.833	0.833	0.810	0.669
DTM		0.819	0.819	0.844	0.773
NPMM-O		0.779	0.779	0.805	0.706
NPMM-G		0.832	0.832	0.854	0.772
DMM		0.589	0.589	0.578	0.424
GSDMM		0.773	0.773	0.681	0.513
OSGM	Ts-T	**0.854**	**0.854**	0.912	0.869
OSGM-ES		0.852	0.852	**0.948**	**0.913**
OSDM		0.842	0.842	0.916	0.864
MF-G		0.850	0.849	0.814	0.661
MF-O		0.823	0.822	0.780	0.628
DTM		0.814	0.814	0.840	0.776
NPMM-O		0.774	0.774	0.766	0.645
NPMM-G		0.826	0.826	0.819	0.711
DMM		0.565	0.565	0.546	0.410
GSDMM		0.785	0.785	0.709	0.560

the number of iterations to 10 and *saved-batches* $= 2$ for MStreamF-G. As defined in [25], we set $\alpha = 0.03$, $\beta = 0.03$, $\delta = 0.03$, and $\epsilon = 3e^{-7}$ for NPMM. For OSDM, we follow the given parameter setting, where $\alpha = 0.002$, $\beta = 0.0004$, and $\lambda = 6e^{-6}$. We select $\alpha = 0.05$, $\beta = 0.004$ and $\alpha = 0.09$, $\beta = 0.006$ for OSGM and OSGM-ES, respectively. We set $\Gamma = 10$ and $\lambda = 1e^{-6}$ for both variants of the proposed algorithm.

E. Comparison to State-of-the-Art Approaches

Here, we compare OSGM with the selecting state-of-the-art algorithms. Table III gives the overall clustering results on different real-world and synthetic data streams. We report NMI, v-measure, Homogeneity, and purity for each algorithm.

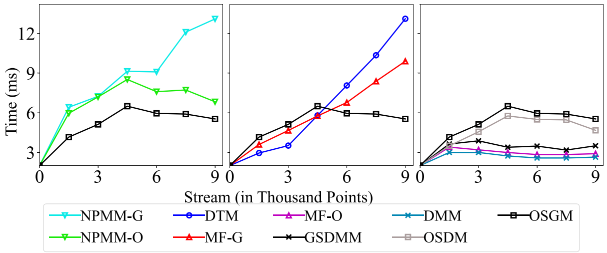


Fig. 5. Runtime in milliseconds (ms) of different text stream clustering algorithms.

The results show the superior performance of the proposed approach with and without semantic smoothing. The analysis of both variations is discussed in Section V-H. In total, OSGM yields the best performance over almost all the datasets. NPMM-G gives the best results in the News real-world dataset, by iterating the entire stream multiple times. However, the online process of the mentioned algorithm does not outperform while comparing it with the reported results of OSGM. Likewise, MF-G also iteratively processes each batch to infer the number of clusters; however, due to the term ambiguity problem, it is unable to achieve high performance. Besides, the crucial part of evaluating the cluster similarity is measured by the homogeneity measure. The significant difference in results on both synthetic data streams proves that OSGM can capture topic-related terms and cluster evolution simultaneously. Additional core term change analysis is provided in the supplementary material due to space limitations.

F. Runtime Analysis

To compare the running time of competing algorithms, we performed all the experiments on a system with core i5-3470 and 8-GB memory. The time consumption of all algorithms on the News-T dataset is shown in Fig. 5, where we split algorithms into three plots to increase the difference visibility. The first plot reflects algorithms having higher execution times than OSGM, which are NPMM-G and NPMM-O. The reason is NPMM exploits pretrained word embeddings to calculate posterior probability. The second plot shows moderate execution time algorithms, which are MF-G and DTM. As we can observe, the execution time of both algorithms abruptly increases with the increase of topics in the stream, which directly affects the runtime of their iterative process for topic inference. The third plot shows the algorithm having comparatively a bit lower execution time than OSGM. The visible difference is due to extra computation required by OSGM to calculate the term co-occurrence matrix for semantic similarity, whereas MF-O, GSDMM, and DMM lack this property. The overall speed of OSGM is more efficient compared to most existing algorithms.

G. Parameter Sensitivity Analysis

We perform a sensitivity analysis for OSGM with respect to four input parameters: 1) α; 2) β; 3) λ; and 4) Γ on the News dataset. Fig. 6(a) shows the effect of α which ranges from $1e^{-3}$ to $1e^{-1}$. It can be observed that the performance

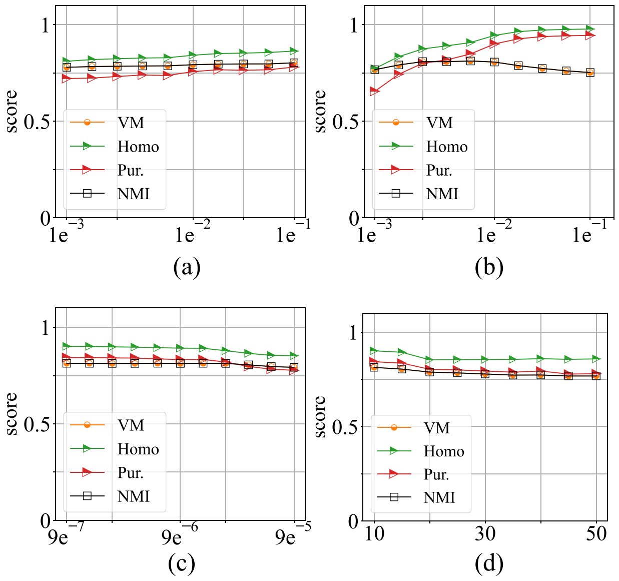


Fig. 6. Parameter sensitivity analysis. (a) α. (b) β. (c) λ. (d) Γ.

of OSGM is quite stable over a large range of α values. The parameter α also contributes to infer the number of clusters, which is why the performance is stable. The multinomial distribution β parameter is responsible while calculating the term distribution similarity. Fig. 6(b) shows the sensitivity analysis over the same range of α. We can observe that after a certain range, the relationship between NMI and Homogeneity becomes inverse. The reason behind this is that the increasing value of β leads toward fine-grain clustering, which directly affects the NMI values. Our model follows the forgetting mechanism on decay factor λ and the clusters are deleted from the model when the value approximately equals 0. Fig. 6(c) depicts the performance of OSGM on different decay factors ranging from $9e^{-7}$ to $9e{-}5$. It can be observed that the behavior of a given evaluation measure is stable and does not show high variation. The triangular time threshold Γ is responsible for capturing evolving features and extracting core terms. Fig. 6(d) shows the sensitivity of a defined threshold ranging from 10 to 50. Interestingly, the performance is stable in terms of homogeneity. However, after a certain range, it starts decreasing, particularly for NMI. This is due to the rapid disappearance of terms in clusters, which lead toward highly fine-grained clustering.

H. Semantic Smoothing Analysis

For a deep understanding of the significance of semantic smoothing, we compared our results of both variants with all state-of-the-art approaches. We can observe that both variants outperformed many state-of-the-art approaches. To explore the importance of introducing semantic smoothing, we demonstrate the parameter sensitivity analysis in terms of multinomial distribution parameter β. Fig. 7(a) depicts the significant difference of semantic smoothing over a wide range of parameters of OSGM and OSGM-ES. It is clearly observable that the variation of NMI over different β values is higher when we do not use semantic smoothing. Likewise, Fig. 7(b) shows semantic smoothing leads toward a high NMI score by following coarse-grain clustering.

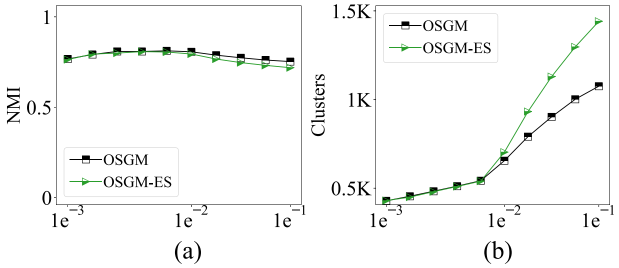


Fig. 7. β sensitivity of OSGM and OSGM-ES on a real dataset in terms of NMI and cluster creation. (a) News. (b) News.

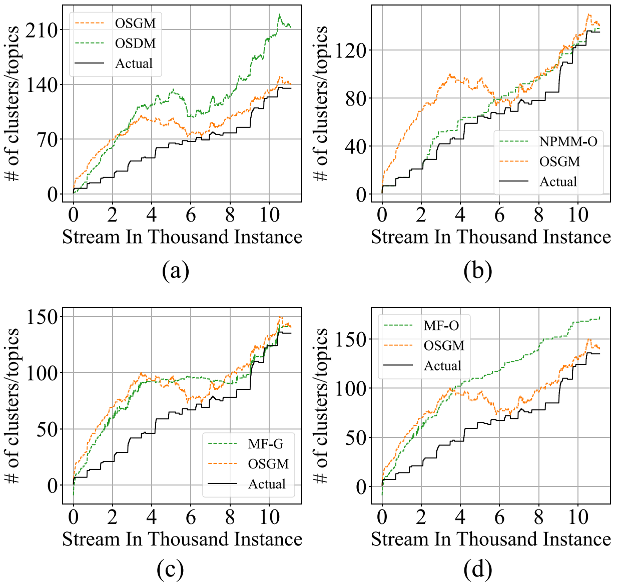


Fig. 8. Predicted versus number of topic comparison with state-of-the-art algorithms. (a) OSDM. (b) NPMM-O. (c) MF-G. (d) MF-O.

I. Topic Estimation Analysis

The OSGM automatically creates clusters based on the calculated probability for the evolving number of topics; thus, it does not need a number of topics as input. Fig. 8 shows the active number of clusters and the actual number of topics over time comparative to different algorithms, including NPMM-O, OSDM, MStreamF-O (MF-O), and MStreamF-G. To interpret the given plots, suppose, our model has $|D|$ number of active documents at time-stamp t, and according to ground truth, D belongs to $\mathcal{L}$ topics. Consider, our model has grouped D into $|M|$ number of clusters. Ideally, $|M| = |\mathcal{L}|$; therefore, to evaluate here in Fig. 8, "Actual" represents $|\mathcal{L}|$ and "OSGM" reflects $|M|$ of the model over time. By observing the model cluster over time, we can conclude that OSGM starts converging toward actual topics as clusters start merging with the passage of timestamps. However, OSDM and MF-O do not infer clusters in any way, which is the reason an increase in clusters is observed continuously. In contrast, clusters of MF-G seem much closer due to the batchwise iterative process. Unlike other algorithms, NPMM uses pretrained word embeddings (which already contains a closer distribution of

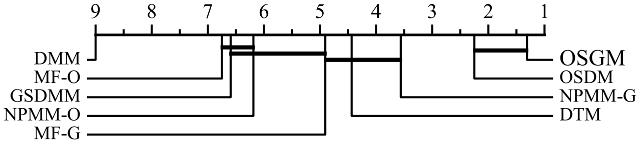


Fig. 9. Nemenyi test on all datasets (real + synthetic) for OSGM.

words); therefore, the number of clusters is much closer to the actual number of clusters. Due to the constant number of topics as the input parameter, we do not include DDM, DTM, and GSDMM, Whereas, NPMM-G is not included as it iteratively processes the entire stream multiple times to infer the number of topics.

J. Statistical Tests

The series of previous state-of-the-art algorithms follows a similar nonparametric statistical model to assess the statistical significance between comparing approaches. As reported in [49], we perform the Friedman rank test with a 95% confidence level. We define the null hypothesis as no statistical difference among competing methods. If the null hypothesis is rejected, we further use the Nemenyi *post-hoc* test to find these differences. The test is conducted on Table III. On the standard threshold $p < 0.05$, for all average ranks of all algorithms on real and synthetic datasets, we obtain p-values $= 5.115e^{-15}$. This confirms that the proposed algorithm statistically differs from other methods. We apply the Nemenyi *post-hoc* test, which enables us to build a critical difference diagram shown in Fig. 9. The closest critical difference score resembles OSGM, which can deal with the evolving topics but cannot deal with the evolving term subspace.

VI. Conclusion

This article proposed a new online semantic-enhanced graphical model for evolving short text stream clustering. Compared to the existing approaches, OSGM does not require to specify the batch size, the dynamic number evolving clusters, and can reduce CFs to extract core terms automatically. It dynamically assigns each arriving document into an existing cluster or generating a new cluster based on the poly urn scheme. More importantly, OSGM incorporates semantic smoothing and term co-occurrence to deal with term ambiguity and coarse-grain clustering in the proposed graphical representation model. By exploiting the triangular time function, the proposed approach can track the change of term distribution over time. Moreover, we further investigated the importance of semantic smoothing in the proposed model. A deep-conducted empirical study on synthetic and real-world datasets further demonstrated the benefits of OSGM compared to many state-of-the-art algorithms.

References

[1] C. Lin, R. Xie, X. Guan, L. Li, and T. Li, “Personalized news recommendation via implicit social experts,” *Inf. Sci.*, vol. 254, pp. 1–18, Jan. 2014.

[2] P. Li *et al.*, “Learning from short text streams with topic drifts,” *IEEE Trans. Cybern.*, vol. 48, no. 9, pp. 2697–2711, Sep. 2018.

[3] K.-Y. Chen, L. Luesukprasert, and S.-C. T. Chou, “Hot topic extraction based on timeline analysis and multidimensional sentence modeling,” *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 8, pp. 1016–1025, Aug. 2007.

[4] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. de Carvalho, and J. Gama, “Data stream clustering: A survey,” *ACM Comput. Surveys*, vol. 46, no. 1, pp. 1–31, 2013.

[5] H. L. Nguyen, Y. K. Woon, and W. K. Ng, “A survey on data stream clustering and classification,” *Knowl. Inf. Syst.*, vol. 45, no. 3, pp. 535–569, 2015.

[6] J. Xuan, J. Lu, G. Zhang, and X. Luo, “Topic model for graph mining,” *IEEE Trans. Cybern.*, vol. 45, no. 12, pp. 2792–2803, Dec. 2015.

[7] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, “A framework for projected clustering of high dimensional data streams,” in *Proc. Int. Conf. Very Large Data Bases*, 2004, pp. 852–863.

[8] L. Jing, M. K. Ng, J. Xu, and J. Z. Huang, “Subspace clustering of text documents with feature weighting K-means algorithm,” in *Advances in Knowledge Discovery and Data Mining*, vol. 3518. Hanoi, Vietnam: Springer, 2005, pp. 802–812.

[9] S. Zhong, “Efficient streaming text clustering,” *Neural Netw.*, vol. 18, nos. 5–6, pp. 790–798, 2005.

[10] N. Sahoo, J. Callan, R. Krishnan, G. Duncan, and R. Padman, “Incremental hierarchical clustering of text documents,” in *Proc. Int. Conf. Inf. Knowl. Manag.*, 2006, p. 357.

[11] C. C. Aggarwal and P. S. Yu, “On clustering massive text and categorical data streams,” *Knowl. Inf. Syst.*, vol. 24, no. 2, pp. 171–196, 2010.

[12] D. M. Blei and J. D. Lafferty, “Dynamic topic models,” in *Proc. ACM Int. Conf. Series*, vol. 148, 2006, pp. 113–120.

[13] Q. Mei and C. X. Zhai, “Discovering evolutionary theme patterns from text—An exploration of temporal text mining,” in *Proc. ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2005, pp. 198–207.

[14] A. Ahmed and E. P. Xing, “Dynamic non-parametric mixture models and the recurrent chinese restaurant process: With applications to evolutionary clustering,” in *Proc. SIAM Int. Conf. Data Min.*, 2008, pp. 219–230.

[15] N. Kawamae, “Trend analysis model: Trend consists of temporal words, topics, and timestamps,” in *Proc. 4th ACM Int. Conf. Web Search Data Min.*, 2011, pp. 317–326.

[16] Y. Wang, E. Agichtein, and M. Benzi, “TM-LDA: Efficient online modeling of latent topic transitions in social media,” in *Proc. Int. Conf. Knowl. Disc. Data Min.*, 2012, pp. 123–131.

[17] R. Huang, G. Yu, Z. Wang, J. Zhang, and L. Shi, “Dirichlet process mixture model for document clustering with feature partition,” *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 8, pp. 1748–1759, Aug. 2013.

[18] J. Yin and J. Wang, “A text clustering algorithm using an online clustering scheme for initialization,” in *Proc. ACM Int. Conf. Knowl. Disc. Data Min.*, 2016, pp. 1995–2004.

[19] H. Gong, T. Sakakini, S. Bhat, and J. Xiong, “Document similarity for texts of varying lengths via hidden topics,” in *Proc. 56th Annu. Meeting Assoc. Comput. Linguist.*, vol. 1, 2018, pp. 2341–2351.

[20] L. Shou, Z. Wang, K. Chen, and G. Chen, “Sumblr continuous summarization of evolving tweet streams,” in *Proc. 36th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2013, pp. 533–542.

[21] S. Liang, E. Yilmaz, and E. Kanoulas, “Dynamic clustering of streaming short documents,” in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2016, pp. 995–1004.

[22] J. Yin, D. Chao, Z. Liu, W. Zhang, X. Yu, and J. Wang, “Model-based clustering of short text streams,” in *Proc. ACM Int. Conf. Knowl. Disc. Data Min.*, 2018, pp. 2634–2642.

[23] A. Hadifar, L. Sterckx, T. Demeester, and C. Develder, “A self-training approach for short text clustering,” in *Proc. 4th Workshop Represent. Learn. NLP (ACL)*, 2019, pp. 194–199.

[24] Z. Haj-Yahia, A. Sieg, and L. A. Deleris, “Towards unsupervised text classification leveraging experts and word embeddings,” in *Proc. Assoc. Comput. Linguist.*, 2019, pp. 371–379.

[25] J. Chen, Z. Gong, and W. Liu, “A nonparametric model for online topic discovery with word embeddings,” *Inf. Sci.*, vol. 504, pp. 32–47, Dec. 2019.

[26] C. Fahy, S. Yang, and M. Gongora, “Ant colony stream clustering: A fast density clustering algorithm for dynamic data streams,” *IEEE Trans. Cybern.*, vol. 49, no. 6, pp. 2215–2228, Jun. 2019.

[27] M. K. Islam, M. M. Ahmed, and K. Z. Zamli, “A buffer-based online clustering for evolving data stream,” *Inf. Sci.*, vol. 489, pp. 113–135, Jul. 2019.

[28] X. Cheng, X. Yan, Y. Lan, and J. Guo, “BTM: Topic modeling over short texts,” *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 12, pp. 2928–2941, Dec. 2014.

[29] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[30] X. Wei, J. Sun, and X. Wang, "Dynamic mixture models for multiple time-series," in *Proc. 20th Int. Joint Conf. Artif. Intell.*, Jan. 2007, pp. 2909–2914.

[31] Y. B. Liu, J. R. Cai, J. Yin, and A. W. C. Fu, "Clustering text data streams," *J. Comput. Sci. Technol.*, vol. 23, no. 1, pp. 112–128, 2008.

[32] H. Amoualian *et al.*, "Streaming-LDA: A Copula-based approach to modeling topic dependencies in document streams," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2016, pp. 695–704.

[33] S. U. Din, J. Shao, J. Kumar, W. Ali, J. Liu, and Y. Ye, "Online reliable semi-supervised learning on evolving data streams," *Inf. Sci.*, vol. 507, pp. 153–171, Jul. 2020.

[34] J. Shao, Y. Tan, L. Gao, Q. Yang, C. Plant, and I. Assent, "Synchronization-based clustering on evolving data stream," *Inf. Sci.*, vol. 501, pp. 573–587, Oct. 2019.

[35] J. Sui, Z. Liu, L. Liu, A. Jung, and X. Li, "Dynamic sparse subspace clustering for evolving high-dimensional data streams," *IEEE Trans. Cybern.*, early access, Nov. 24, 2020, doi: 10.1109/TCYB.2020.3023973.

[36] Y. Jianhua and J. Wang, "A model-based approach for text clustering with outlier detection," in *Proc. 32nd IEEE Int. Conf. Data Eng.*, 2016, pp. 625–636.

[37] Y. Liu, J. Cai, J. Yin, and A. W.-C. Fu, "Clustering massive text data streams by semantic smoothing model," in 3rd Int. Conf. Adv. Data Mining Appl., 2007, pp. 389–400.

[38] C. C. Aggarwal *et al.*, "A framework for clustering evolving data streams," in *Proc. Int. Conf. Very Large Data Bases*, 2003, pp. 81–92.

[39] D. Blackwell and J. B. MacQueen, "Ferguson distributions via Pólya urn schemes," *Ann. Stat.*, vol. 1, no. 2, pp. 353–355, 1973.

[40] T. S. Ferguson, "A Bayesian analysis of some nonparametric problems," *Ann. Stat.*, vol. 1, no. 2, pp. 209–230, 1973.

[41] R. M. Neal, "Markov chain sampling methods for Dirichlet process mixture models," *J. Comput. Graph. Stat.*, vol. 9, no. 2, pp. 249–265, 2000.

[42] S. U. Din and J. Shao, "Exploiting evolving micro-clusters for data stream classification with emerging class detection," *Inf. Sci.*, vol. 507, pp. 404–420, Jan. 2020.

[43] G. Huang *et al.*, "Mining streams of short text for analysis of world-wide event evolutions," *World Wide Web*, vol. 18, no. 5, pp. 1201–1217, 2015.

[44] M. Tennant, F. Stahl, O. Rana, and J. B. Gomes, "Scalable real-time classification of data streams with concept drift," *Future Gener. Comput. Syst.*, vol. 75, pp. 187–199, Oct. 2017.

[45] 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.

[46] 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.

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

[48] J. Kumar, J. Shao, S. Uddin, and W. Ali, "An online semantic-enhanced dirichlet model for short text stream clustering," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguist.*, 2020, pp. 766–776.

[49] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006.

Jay Kumar received the master's degree from Quaid-i-Azam University, Islamabad, Pakistan, in 2018. He is currently pursuing the Ph.D. degree with the University of Electronics Science and Technology of China, Chengdu, China.

He is working with Data Mining Lab, School of Computer Science and Engineering, University of Electronics Science and Technology of China. His current research work has been published in top conference of ACL, IEEE Transactions on Cybernetics, and *Information Sciences*. His main interest of research include data stream mining and natural language processing.

Salah Ud Din received the Ph.D. degree from the School of Computer Science and Engineering, the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2020.

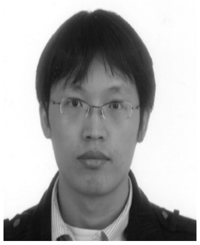
He is currently working with the School of Computer Science and Engineering, UESTC, also with the Yangtze Delta Region Institute (Huzhou), UESTC, Huzhou, China, and also with the Department of Computer, COMSATS University Islamabad (Abbottabad Campus), Islamabad, Pakistan. His research interests focus on data stream mining, especially on data stream classification, novel class detection, and semisupervised learning. Other related interests include machine learning, data science, big data, text mining, pattern recognition, image processing, and document image analysis.

Qinli Yang received the Ph.D. degree from the University of Edinburgh, Edinburgh, U.K., in 2011.

She is currently working with the University of Electronic Science and Technology of China, Chengdu, China. Her recent research interests focus on data mining driven water resources research. She has published many papers in prestigious journals, such as *Water Research*, *Environmental Modeling and Software*, and *Journal of Environmental Management*, as well as several papers in the field of data mining.

Rajesh Kumar received the Ph.D. degree in computer science and engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2019.

He is currently working as an Assistant Researcher with the Yangtze Delta Region Institute (Huzhou), UESTC, Huzhou, China. In addition, he has published more than 20 articles in various International journals and conferences. His research interests include machine learning, deep leaning, malware detection, Internet of Things, and blockchain technology.



Junming Shao (Member, IEEE) received the Ph.D. degree (Highest Hons., *summa cum laude*) from the University of Munich, Munich, Germany, in 2011.

He is currently a Professor of Computer Science with the University of Electronic Science and Technology of China, Chengdu, China. He not only published papers on top-level data mining conferences, such as KDD, WWW, ACL, IJCAI, IEEE/TKDE but also published data mining-related interdisciplinary work in leading journals, including *Brain*, *Neurobiology of Aging*, and *Water Research*. His research interests include data mining and machine learning, especially for subspace clustering, graph mining, and data stream mining.

Prof. Shao became the Alexander von Humboldt Fellow in 2012.