#Manuscript: CYB-E-2020-12-3186: An Online Semantic-enhanced Graphical Model for Evolving Short Text Stream Clustering

Summary of Changes

Thank you very much for the reviewers' valuable comments on our manuscript. We have again revised the manuscript according to your kind advice and constructive suggestions. Enclosed, please find a detailed response to reviewers. Generally, we have made the following major changes.

1. Section III is revised by adding model generative process.
2. Detailed description of equations, especially triangular time decay function, are given with examples.
3. Section IV is revised and reorganized to give more clear understanding.
4. Mathematical definitions of all evaluation measures are added.
5. We have added more figures of runtime complexity comparison with SOTA.
6. Algorithms and their respective notations are revised and reorganized for more clear understanding.
7. We have added more figures to compare cluster generation among SOTA.
8. Detailed description related to construction of dataset are revised.
9. We carefully check the whole manuscript, and the grammatical mistakes and typos are corrected.

For details, please refer to the "Response to Reviews" as follows.

Response to Reviews

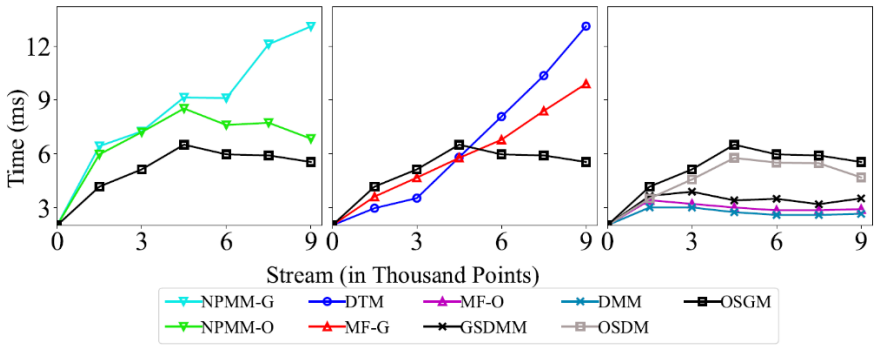
We are very thankful for the reviewers for spending a substantial amount of time to look over the paper and providing valuable comments. We have improved the manuscript accordingly. In the following, we give a point-by-point response to your concerns.

Associate Editor comments for Authors:

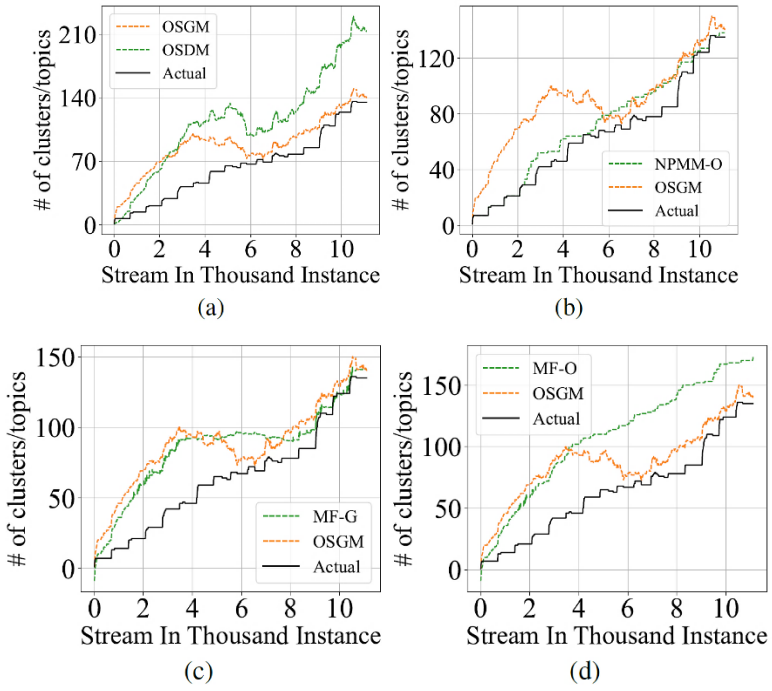
The revised version needs to provide more interpretation and analysis on the results..

Reply: Thank you very much for the valuable comment. We have revised the manuscript according to the reviewers' comments and suggestions.

We have split one plot into three plots to conclude the computational complexity of proposed algorithm in the revised manuscript (**see Page: 10, Section V**). To compare the running time of competing algorithms, the time consumption of all algorithms on 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 time than OSGM, which are NPMM-G and NPMM-O. The reason is NPMM exploit pre-trained word embeddings to calculate posterior probability. The second plot shows moderate execution time algorithms which are MF-G and DTM. As we can clearly observe that both algorithms exucution time abruptly increase as topics in stream increase, which directly effects the runtime of their iterative process for topic inference. The third plot shows algorithm having comparative a bit lower execution time than OSGM. The visible difference is due to extra computation required by OSGM to calculate 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 with most existing algorithms.



We also added clusters creation comparison with SOTA to provide deep insight of proposed model (**see Page: 11, Section V**). The OSGM automatically creates clusters based on calculated probability for evolving number of topics, thus it does not need number of topics as input. Figure below shows the active number of clusters and actual number of topics over time comparative to different algorithms including NPMM-O, OSDM, MStreamF-O (MF-O), and MStreamF-G. To interprets the given plots, suppose, our model has |D| number of active documents at time-stamp t, and according to ground truth D belongs to L topics. Consider, our model has grouped D into |M| number of clusters. Ideally, |M| = |L|, therefore to evaluate here in given figure the ”Actual” represents the |L| and ”OSGM” reflects the |M| of the model over time. By observing the model cluster over time we can conclude that OSGM starts converging towards actual topics as clusters start merging with the passage of timestamps. However, OSDM and MF-O does not infer clusters in any way, that is the reason increase in clusters is observed continuously. In contrast, clusters of MF-G seems much closer due to batch-wise iterative process. Unlike other algorithms, NPMM uses pretrained word embeddings (which already contains a closer distribution of words), therefore the number of clusters are much closer to actual number of clusters. Due to constant number of topics as input parameter, we do not include DDM, DTM, and GSDMM. Whereas, NPMM-G is not included as it iteratively process whole stream multiple times to infer the number of topics.



Referee: 1

Recommendation: **Accept**

Reply: Thank you very much for your valuable evaluation and accepting our manuscript.

Referee: 2

Recommendation: **Accept With Minor Changes**

Reply: Thank you very much for comments and accepting our research work. We address each comment and respective improvement as follows.

(1) The Lack of necessary analyses. In experiments, some experimental results have been obtained, but why? The reasons are required..

Reply: We have revised results section and provide the reason for each analysis.

- Cluster Creation Analysis: To analyze the actual number of topics with created cluster by model, we perform cluster creation analysis. The series of topic models focus on approximating number of latent topics in the given corpus. The more clusters an algorithm generates more it compromises with its robustness.

  The OSGM automatically creates clusters based on calculated probability for evolving number of topics, thus it does not need number of topics as input. Figure below shows the active number of clusters and actual number of topics over time comparative to different algorithms including NPMM-O, OSDM, MStreamF-O (MF-O), and MStreamF-G. To interprets the given plots, suppose, our model has |D| number of active documents at time-stamp t, and according to ground truth D belongs to L topics. Consider, our model has grouped D into |M| number of clusters. Ideally, |M| = |L|, therefore to evaluate here in given figure the ”Actual” represents the |L| and ”OSGM” reflects the |M| of the model over time. By observing the model cluster over time we can conclude that OSGM starts converging towards actual topics as

clusters start merging with the passage of timestamps. However, OSDM and MF-O does not infer clusters in any way, that is the reason increase in clusters is observed continuously. In contrast, clusters of MF-G seems much closer due to batch-wise iterative process. Unlike other algorithms, NPMM uses pretrained word embeddings (which already contains a closer distribution of words), therefore the number of clusters are much closer to actual number of clusters. Due to constant number of topics as input parameter, we do not include DDM, DTM, and GSDMM. Whereas, NPMM-G is not included as it iteratively processes whole stream multiple times to infer the number of topics.

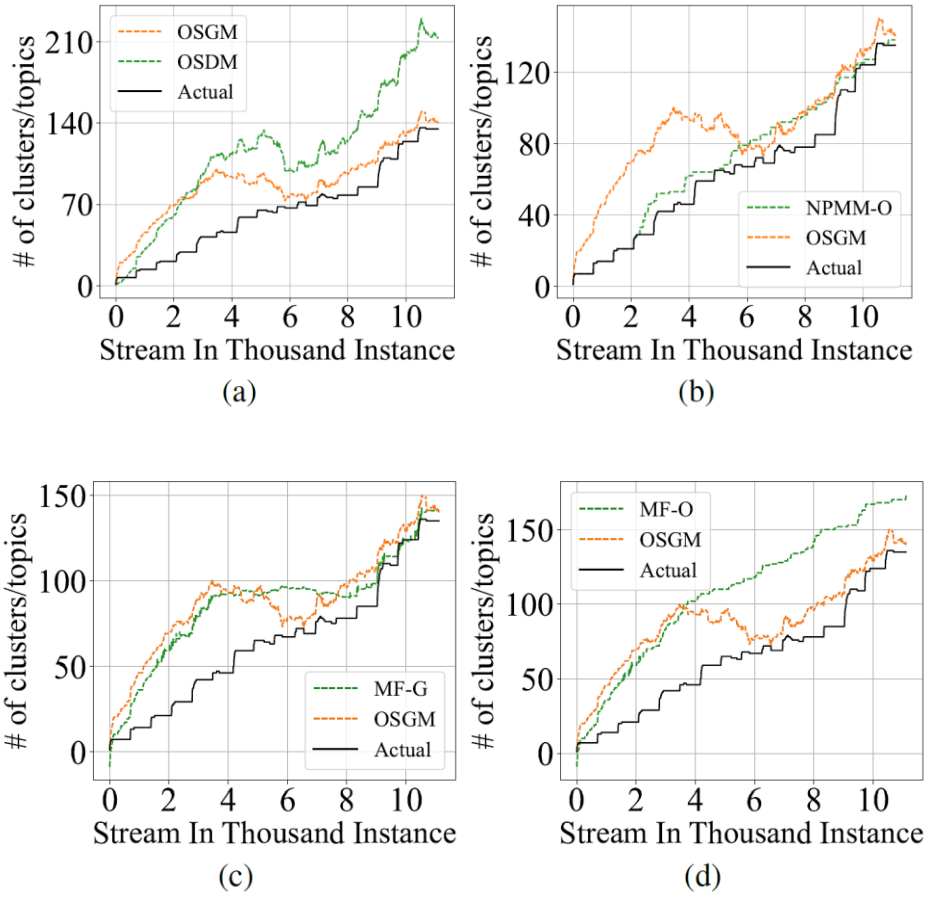


Figure 1 Cluster creation comparison with actual number of topics

- Runtime: Low computation time is among a vital constraint while designing a model for processing streaming data. Less computational time motivates algorithm to work

on high and low velocity data streams. Therefore, we perform the runtime complexity evaluation of our proposed model. The chosen state-of-the-art algorithms either (i) specifically designed for stream environment such as NPMM, and MStreamF, or (ii) static topic modelling algorithms such as GSDMM, DTM and DMM. The static topic modeling approaches required a pre-defined number of topics to cluster the data. Whereas, streaming algorithms do not require a constant number of topics as input. Therefore, we compare runtime complexity of our model with only streaming algorithms.

(2) *Authors addressed the proposed method can detect topic evolution and concept drift. What are concept drifts in the experiments? We know that there are benchmark evaluation measures for concept drifting detection, such as delay, missing concepts, false alarm. How is the performance of the proposed method in this case?*

Reply: Thank you for your valuable comment. We would like to give a deep insight about concept drift in text streams. Generally, the concept drift usually refers to change in the distribution. In context of online topic modeling, it refers to following challenges:

(i) **Unknow number of topics:** In this context we can define this problem as **novel class detection** in which several new classes (here we refer as *Topic*) may arrive at any time in stream. As shown in figure below.

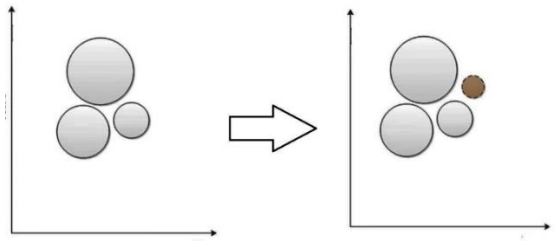


Figure 2 New Class Arrival

In our experiments, as an example of synthetic datasets News-T and Tweets-T,

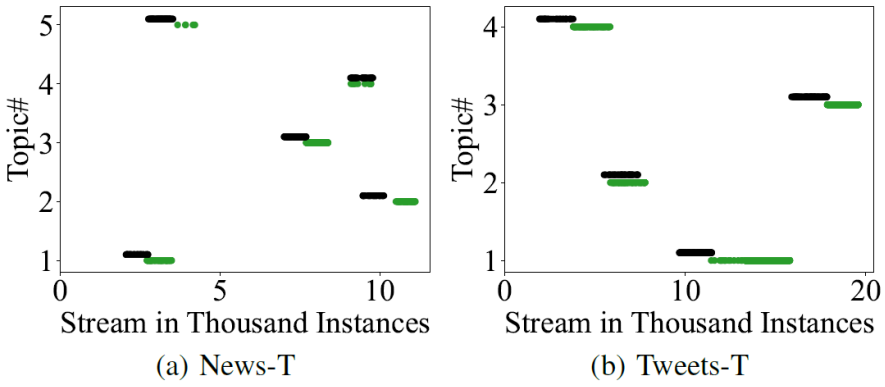


Figure 3 Instances of Topics containing term distribution change

where we do not know how many topics will arrive at time t , therefore, we preformed experiments and show results on different evaluation measures.

(ii) **Change in core terms of Topic:** In many cases, especially in social media short text streams, the core terms of a single topic may change at any time, therefore, we can call this problem as feature evolution. As shown in figure below.

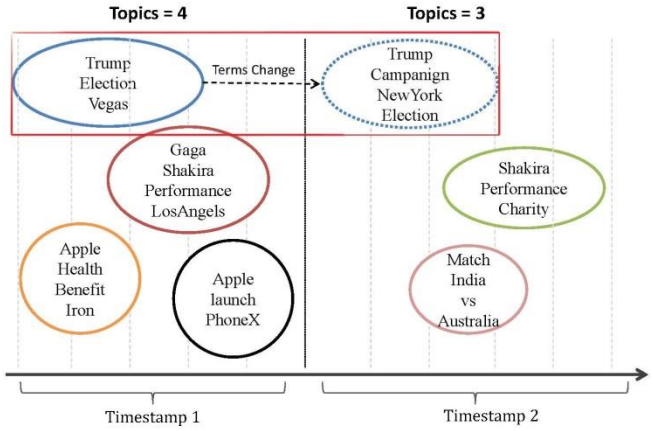


Figure 4 An example containing Three challenges of Text stream topic modeling

For this reason, we track a topic, whose change in core terms are given in figure 5 below, and we track core terms of same topic in OSGM over four different timestamps to assure that our model can capture change in core terms, depicted in Figure 6 below.

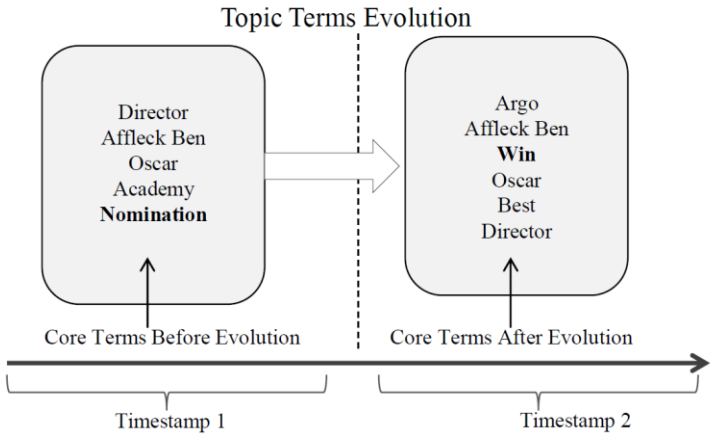


Figure 5 Change in Core Terms of a Topic in News dataset

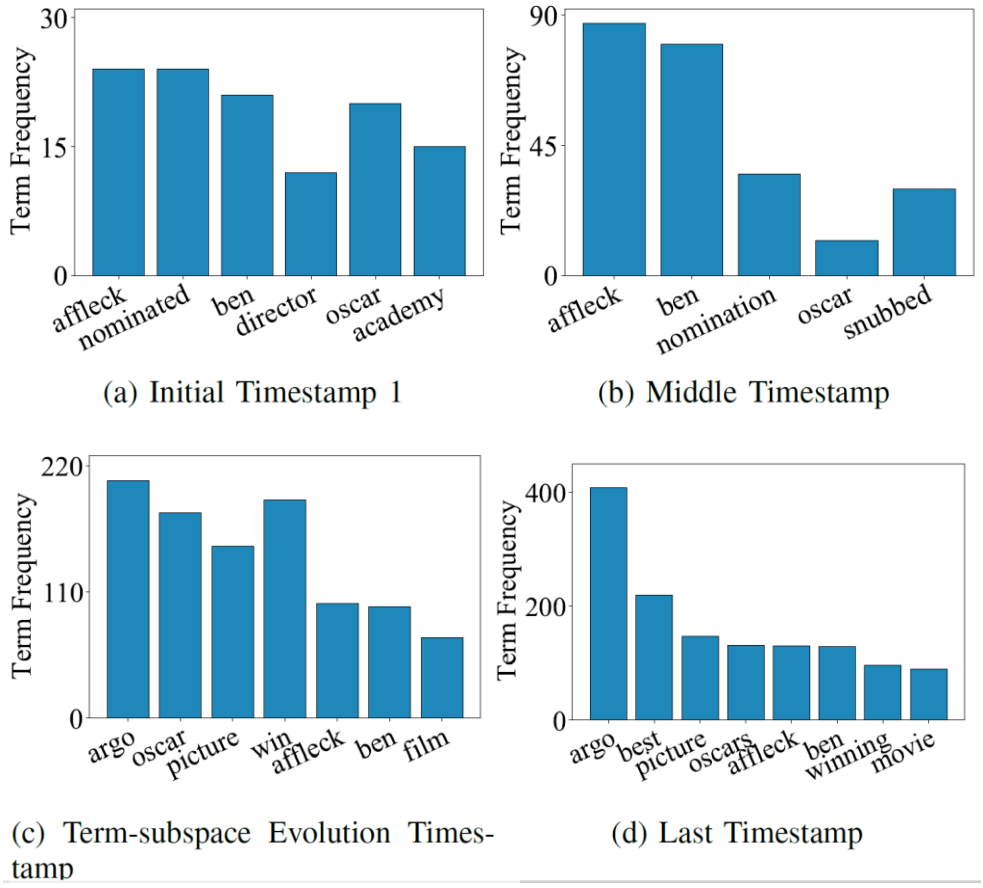


Figure 6 Change in Core Terms by OSGM

Due to space limitation, we add this analysis in *Supplementary materials*.

(iii) Distributional Change in Topic-related Terms: In many cases where topic related terms do not change over time, in contrast their occurrence change. The score used to calculate the similarity is dependent on their frequency. Over the active period of time span of a topic, the term distribution may change and which leads to change in cluster boundary. A example is show in figure below.

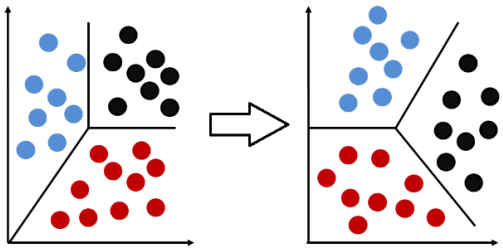


Figure 7 Example of class boundary change

We choose a topic as a case study to exploit the cluster term-subspace evolution. The given below figures show the actual evolution of terms over time. We track the distribution of core terms captured by OSGM related to the same topic at four different timestamps, as depicted in previous mentioned Figure 6 above. It is observed that OSGM successfully traces the term evolution and can extract the core terms.

(3) *There are several grammar errors*

Reply: Thank you for mentioning the grammar errors. We have carefully revised the manuscript and removed all typos and grammatical errors.

Referee: 3

Recommendation: Reject & Resubmit

Reply: Thank you very much for your very useful comments. We address each comment and respective improvement as follows.

1) *The authors should include a generative procedure for the proposed OSGM. I believe that the authors can find many examples in the related literature.*

Reply: Thank you. We have added the generative process (**see Page: 5, Section IV**). The generative process for topic generation in stream is presented as follow:

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. $w_d \sim Multinomial(\mathcal{N}_{t,z}|z_d)$

At time t, the distribution θ_t is sampled from Dirichlet process with concentration parameter α and previous distribution θ_{t-1}. For each document in stream, we sample distribution for each topic and for each word in document we draw its distribution from multinomial distribution.

2) *Equation 9 should be discussed in with some example to support better understanding. Right now, I do not see the meaning of this formulation.*

Reply: Thank you very much. We have added an example for better understanding (**see Page: 6, Section IV**).

To highlight core terms and term subspace change in online way, we first maintain arriving timestamps of each term. We then calculate age of the cluster by adopting triangular number of its size, defined as,

$$Triangular\ Number\ \Delta f(T) = ((T^2 + T)/2)$$

Here, T is the timestamp number. We then measure term recency score by taking ratio of the sum of its arrivals and cluster age. For example, cluster z contains 9 documents then the age of cluster z will be:

$$Age_z = \Delta f(9) - \Delta(1) = ((9^2 + 9)/2) - ((1^2 + 1)/2)$$

Let us suppose D_1, D_2, and D_8 belong to z which are:

D_1: "entry performance of Lady gaga."

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

D_2: "Gaga and Shakira rocked!"

Their arrival timestamps are 1, 2, and 8, respectively. The arrivals of a terms (ta) in cluster is stored as:

$$ta_{gaga} = \{timestamp: 1, \quad timestamp: 2, \quad timestamp: 8\}$$

and the sum of time arrivals of a term is,

$$sum(ta_{gaga}) = \sum_{t \in ta_{gaga}} t$$

To calculate the recency score of '*gaga'*, which will be

$$recency_{gaga} = \big((1 + 2 + 8) \times 100\big)/Age_z$$

If the recency score is less than our defined threshold Γ then the term will be deleted from cluster as it will be considered as outdated.

3) *In algorithm 2, what S_z refers to? what $arrival_j$ refers to?*

Reply: Thank you so much. We have corrected the symbol mistake and we revised symbols for better understanding (**see Page: 7, Section IV**). Here, S_z refers to starting timestamp of cluster. $arrival_j$ referred to ta_w which stores the arrival timestamps for a word in cluster.

Algorithm 2: Update Cluster Term-subspace

```
/* Update weight of terms in each
   active cluster                    */
Function updateTermsSubspace (Γ, M)
    S_z = Δf(1) using Equation (9)
    foreach z ∈ M do
        E_z = Δf(m_z) using Equation (9)
        Age_z = E_z − S_z + 1
        foreach (w, ta_w) ∈ ta_z do
            // sum arrival timestamps of w
            sumta_w = Σ_{t∈ta_w} t
            recency_w = (sumta_w × 100)/Age_z
            if recency_w < Γ then
                remove(w, z)
            end
        end
    end
    return M
```

4) *In Equation 10, what Δt refers to? how to calculate this term?*

Reply: Thank you for mentioning. We have revised the subsection and provided the definition of Δt (**see Page: 7, Section IV**). The Δt is the difference of time from cluster created to current timestamp of model. We define Δt as,

$$\boldsymbol{\Delta t = u_z - t_c}$$

Here, u_z is cluster creation timestamp, and t_c is current timestamp. The purpose of Δt is to measure how much old this cluster is. On the basis of time duration we check either this cluster has been updated over the time or not. If not then the decay function weight will be reduced close to zero, which means that model needs to remove this cluster as it cannot capture the current concept. We revise the formulation for clear understanding of decay function as,

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

5) *In algorithm 3, line 12 for F_{Z_i} calculation is problematic. There is no index i in the right part.*

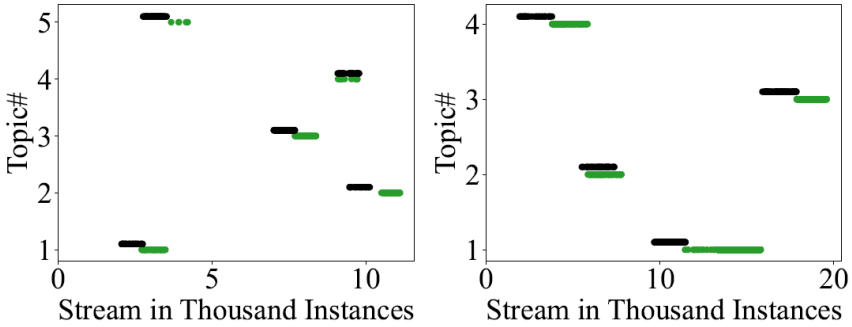
Reply: Thank you very much. We have revised the notations of Algorithm 3 for clear understanding (**see Page: 7, Section IV**).

Algorithm 3: Update Active Clusters

```
/* Update weight of all active
   clusters                                   */
Function updateActiveClusters(λ, M)
    Old = φ
    foreach z ∈ M do
        calculate l_z using Equation (10)
        if l_z ≈ 0 then
            Old = Old ∪ z
        end
    end
    foreach z_i ∈ Old do
        foreach z_act ∈ (M − Old) do
            if V_{z_act} ∩ V_{z_i} ≠ φ then
                P_{Z_i} = p(z_d = z_act, z_i) using Equation
                  (5)
            end
        end
        max = arg max_i (P_{Z_i})
        P_{Z_n} = p(z_d = z_new, z_i) using Equation (8)
        if P_{Z_max} < P_{Z_n} then
            M = M − z_i
        else
            merge(z_i, Z_max) using Definition 1
        end
    end
return M
End Function
```

6) *Figure 3 shows the change in term distribution. But it is hard to interpret y-axis? how to measure the change in a distribution with a scalar? It is better that the authors illustrate this point in detail.*

Reply: Thank you for mentioning. For a clear visibility and understanding, we have improved the figures by adding y-axis and revised the caption (**see Page: 8, Section V**). We also revised the description of these figure in synthetic dataset construction subsection.



(a) News-T (b) Tweets-T

In News-T and Tweets-T dataset, we analyzed some topics whose term distribution change in sub-topics over time. The given figures show the **arrival of instances** of sub-topics over stream. Here, black points show the instances containing initial distribution of terms related to a new topic, whereas, the green points depict instances in which change in term distribution (concept drift) occur.

7) *It is more self-contained to include the definition of each evaluation metric in math language*.

Reply: Thank you very much. We have added mathematical definition of each evaluation measure to make it more self-contained (**see Page: 8, Section V**).

We adopted four highly used evaluation metrics for deep analysis of all algorithms, which include Purity (Pur), VMeasure (V-M), Homogeneity (Homo), and Normalized Mutual Information (NMI). The purity of cluster $z \in Z$ is defined by,

$$purity(z) = \frac{1}{|z|}\max(n_{c,z}|c \in C)$$

Here, $|z|$ is the size of cluster, C reflects all classes and $n_{c,z}$ reflects the number of instances in z related to class c. Homogeneity measures that each cluster should have only members of a single class, defined as:

$$Homo = 1 - H(C|Z) \times H(C)$$

Here, $H(C|Z)$ is the conditional entropy and $H(C)$ represent the coverage of classes, defined as,

$$H(C|Z) = \sum_{z,c} \frac{n_{c,z}}{N} . \log \left(\frac{n_{c,z}}{\sum_c n_{c,z}}\right)$$

$$H(C) = \sum_{c} \frac{\sum_z n_{c,z}}{N} . \log \left(\frac{\sum_z n_{c,z}}{N}\right)$$

Here, $n_{c,z}$ is the number of documents exist in c as well as in z, the total number of documents in is represented by N. The V-measure calculates how successfully the criteria of completeness and homogeneity are satisfied, defined as,

$$VMeasure = \frac{2(Homo \times Co)}{Homo + Co}$$

$$Co = 1 - H(Z|C) \times H(Z)$$

$$H(Z|C) = \sum_{c,z} \frac{n_{c,z}}{N} . \log \left(\frac{n_{c,z}}{\sum_z n_{c,z}}\right)$$

$$H(Z) = \sum_{z} \frac{\sum_c n_{c,z}}{N} . \log \left(\frac{\sum_c n_{c,z}}{N}\right)$$

The NMI measure calculates the overall clustering quality, defined as,

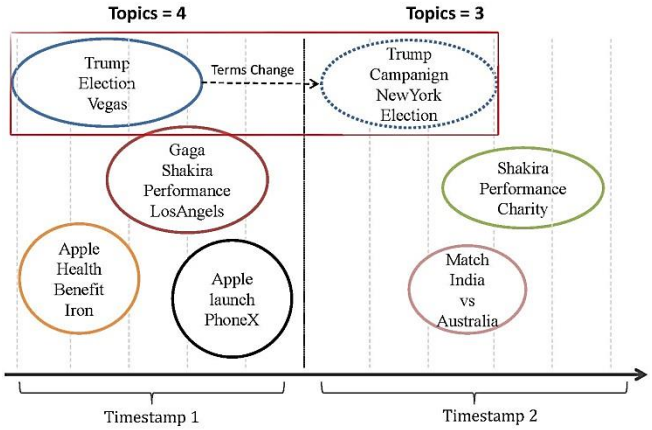
$$NMI = \frac{\sum_{c,z} n_{c,z} . \log \left(N . \frac{n_{c,z}}{n_c . n_z}\right)}{\sqrt{\left(\sum_c n_c . \log \left(\frac{n_c}{N}\right)\right)\left(\sum_z n_z . \log \left(\frac{n_z}{N}\right)\right)}}$$

Here, n_z is the number of documents in cluster z, and n_c is number of document of class c.

.

8) In Figure 8 and Figure 10, why we need to display the term frequency? what information does this term frequency convey? is it better that we use some relative measure to reflect the change?

Reply: Thank you. We have reorganized the Figure 8 and Figure 10. Generally, the concept drift usually refers to change in the distribution of any kind. As shown in Figure,



we have depicted an example that how core terms of topics or their distribution may change over time. To evaluate this, we track a topic from News-T in OSGM, where we show how our approach maintain core terms of same topic over time. The key part of this analysis is to evaluate the effect of triangular time decay function to remove unnecessary terms from cluster. Therefore, we show the high frequent terms in a cluster over different timestamps.

Due to space limitation we reorganized this analysis and added in *Supplementary material* in revised manuscript.

9) *According to Figure 11, it is hard to draw a conclusion that "the overall speed of OSGM is more efficient compared with most existing algorithms".*

Reply: Thank you very much for the valuable comment. We have split the one plot into three plots to conclude the computational complexity of proposed algorithm in the revised manuscript (**see Page: 10, Section V**).

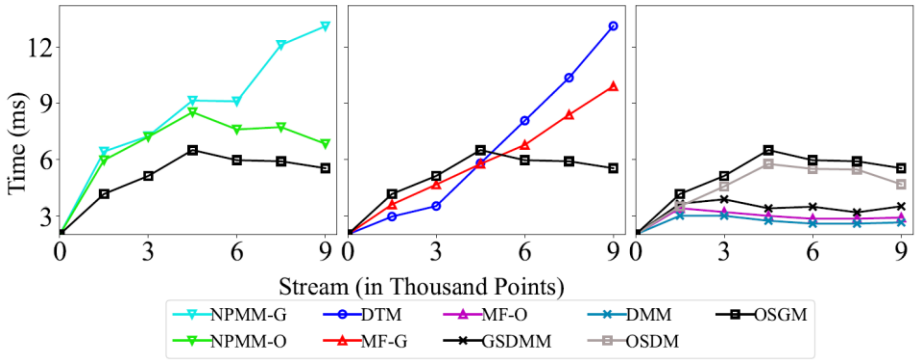


Figure 1 Running Time Comparison of Competing Algorithms

The time consumption of all algorithms on News-T dataset is shown in Figure, where we split algorithms into three plots to increase the difference visibility. The first plot reflects algorithms having higher execution time than OSGM, which are NPMM-G and NPMM-O. The reason is NPMM exploit pre-trained word embeddings to calculate posterior probability. The second plot shows moderate execution time algorithms which are MF-G and DTM. As we can clearly observe that both algorithms execution time abruptly increase as topics in stream increase, which directly effects the runtime of their iterative process for topic inference. The third plot shows algorithm having comparative a bit lower execution time than OSGM. The visible difference is due to extra computation required by OSGM to calculate 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 with most existing algorithms.

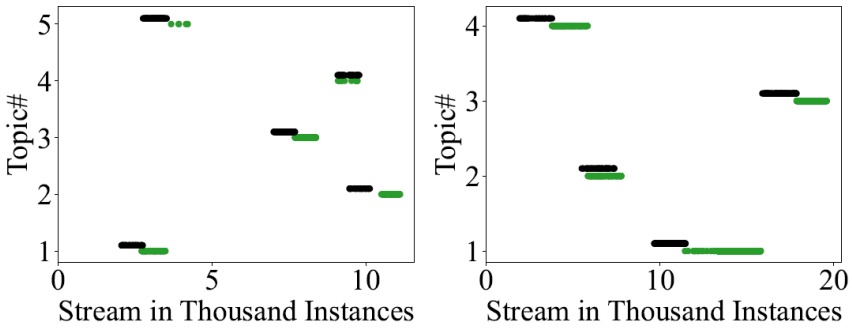
Referee: 4

Recommendation: **Accept With Minor Changes**

Reply: Thank you very much for comments and accepting our research work. We address each comment and respective improvement as follows.

1) *In Figure 4, it is recommended to give the meaning of the vertical axis of each graph ".*

Reply: Thank you for mentioning. For a clear visibility and understanding, we have improved the figures by adding y-axis and revised the caption (**see Page: 8, Section V**). We also revised the description of these figure in synthetic dataset construction subsection.



(a) News-T (b) Tweets-T

Naturally, we may find a situation where topics in social media appear only for a certain period and then disappear. The next thing to observe that in many situations the distribution of topic related terms changes. We constructed synthetic dataset by injecting two mentioned conditions in News and Tweets dataset. We analyzed some topics whose term distribution change in sub-topics over time. The given figures show the **arrival of instances** of sub-topics over stream. Here, black points show the instances containing initial distribution of terms related to a new topic, whereas, the green points depict instances in which change in term distribution (concept drift) occur.

2) *It is recommended to give an explanation for the statistics of the datasets reported in Table II, especially the formula symbols involved in the table.*

Reply: Thank you very much for mentioning. We have revised the caption and description of table to give my clear reading.

TABLE II: Statistics of Each dataset where Ds represent name of dataset, $|D|$ represents total number of documents, $|V|$ is number of unique terms, $|T|$ is total number of topics, and $\mathcal{L}$ is average document length.

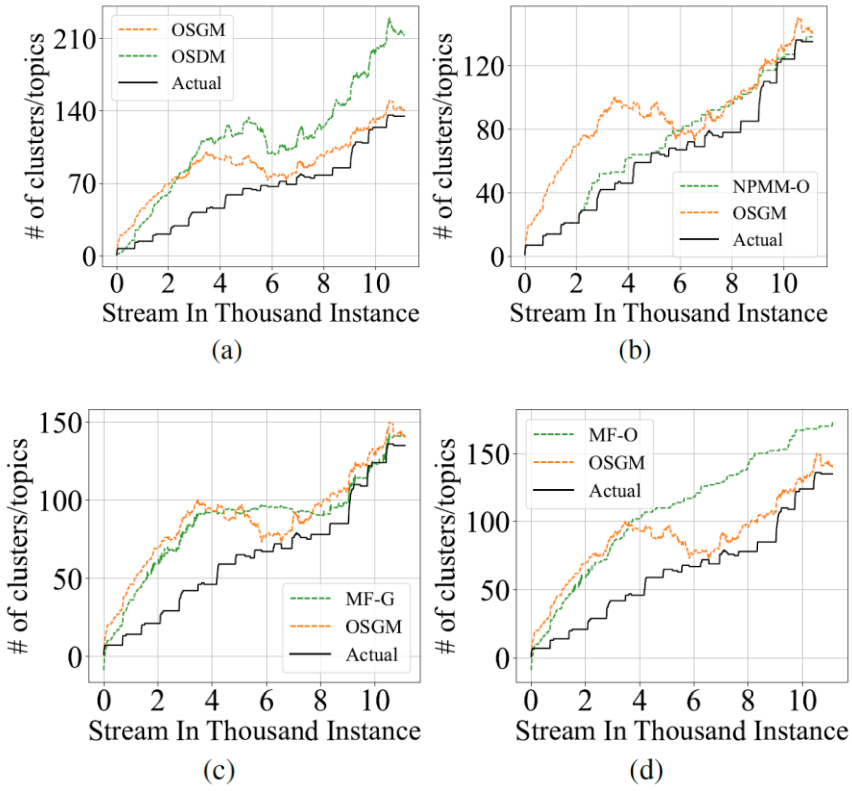
Ds	$\|D\|$	$\|V\|$	$\|T\|$	$\mathcal{L}$
News	11109	8110	152	6.23
Tweets	30322	12301	269	7.97
News-T	11109	8110	147	6.23
Tweets-T	30322	12301	265	7.97

Here, both datasets contain less than 10 average document length ($\mathcal{L}$) which assure datasets fit for short text analysis. Specifically, the News dataset contains lower average document length , compared with Tweets dataset, because the dataset was constructed by collecting headlines of news streams.

3) *It is recommended to compare the number of clusters/topics predicted by different methods with the actual number of clusters/topics in Figure 6, in order to fully verify the proposed method.*

Reply: Thank you for your valuable comment. The series of topic models focus on approximating number of latent topics in the given corpus. The more clusters an algorithm generates more it compromises with its robustness. Therefore, to analyze the actual number of topics with created cluster by model, we perform cluster creation analysis. The OSGM automatically creates clusters based on calculated probability for evolving number of topics, thus it does not need number of topics as input. Figure below shows the active number of clusters and actual number of topics over time comparative to different algorithms including NPMM-O, OSDM, MStreamF-O (MF-O), and MStreamF-G. Due to constant number of topics as input parameter, we do not include DDM, DTM,

and GSDMM. Whereas, NPMM-G is not included as it iteratively process whole stream multiple times to infer the number of topics.



To interprets the given plots, suppose, our model has |D| number of active documents at time-stamp t, and according to ground truth D belongs to L topics. Consider, our model has grouped D into |M| number of clusters. Ideally, |M| = |L|, therefore to evaluate here in given figure the ”Actual” represents the |L| and ”OSGM” reflects the |M| of the model over time. By observing the model cluster over time we can conclude that OSGM starts converging towards actual topics as clusters start merging with the passage of timestamps. However, OSDM and MF-O does not infer clusters in any way, that is the reason increase in clusters is observed continuously. In contrast, clusters of MF-G seems much closer due to batch-wise iterative process. Unlike other algorithms, NPMM uses pretrained word embeddings (which already contains a closer distribution of words), therefore the number of clusters are much closer to actual number of clusters.

Referee: 5

Recommendation: **Accept With Minor Changes**

Reply: Thank you very much for comments and accepting our research work. We address each comment and respective improvement as follows.

1) *Explain the differences and relationships between the proposed method OSGM in this paper and the OSDM.*

Reply: Thank you. We have considered the suggestion in revised manuscript. As well as we discuss major and subtle differences between two models in following table to provide a more clear insight.

OSGM	OSDM
1. track the evolution of core terms for each topic.	1. Do not track change in terms for each topic.
2. Remove outdated terms of each cluster using Triangular time decay $Triangular\ Number\ \Delta f(T) = ((T^2 + T)/2)$	2. New terms can be added into cluster but terms cannot be removed. Due to large size of cluster, the cluster contain large number of non-useful terms which directly effects the homogeneity using β parameter.
$$p\left(z_d = z|\vec{z}, \vec{d}, \alpha, \beta\right) = \left(\frac{m_z}{D-1+\alpha D}\right) \cdot \left(\frac{\prod_{w\in d}\prod_{j=1}^{N_d^w}(n_z^w \cdot lCF_w) + \beta + j - 1}{\prod_{i=1}^{N_d} n_z + (\beta V_{z\cup d}) + i - 1}\right) \cdot \left(1 + \sum_{w_i\in d \wedge w_j \in d} cw_{ij}\right)$$ While calculating the cluster-document probability, it normalizes the probability by considering term space of document and cluster.	$$p\left(z_d = z|\vec{z}, \vec{d}, \alpha, \beta\right) = \left(\frac{m_z}{D-1+\alpha D}\right) \cdot \left(\frac{\prod_{w\in d}\prod_{j=1}^{N_d^w}(n_z^w \cdot lCF_w) + \beta + j - 1}{\prod_{i=1}^{N_d} n_z + V\beta + i - 1}\right) \cdot \left(1 + \sum_{w_i\in d \wedge w_j \in d} cw_{ij}\right) \quad (3)$$ It normalizes the probability by considering term space of model.

$p\left(z_d = z|\vec{z}_{\neg d}, \vec{d}, \alpha, \beta\right) = \left(\frac{\alpha D}{D-1+\alpha D}\right) \cdot \left(\frac{\prod_{w\in d}\prod_{j=1}^{N_d^w}\beta + j - 1}{\prod_{i=1}^{N_d}(\bar{V}_z\beta) + i - 1}\right)$ Likewise, to calculate probability for new cluster, OSGM consider average size of cluster vocabulary while normalization to avoid the term scarcity of model.	$p\left(z_d = z|\vec{z}_{\neg d}, \vec{d}, \alpha, \beta\right) = \left(\frac{\alpha D}{D-1+\alpha D}\right) \cdot \left(\frac{\prod_{w\in d}\prod_{j=1}^{N_d^w}\beta + j - 1}{\prod_{i=1}^{N_d}(V\beta) + i - 1}\right)$ Likewise, it consider global vocabulary to compute the probability for new cluster.
5. To reduce the cluster scarcity and make model more robust, OSGM contains cluster **merging process** of outdated cluster. So that cluster of same topic can merge.	5. OSDM directly deletes the cluster from model by calculating exponential decay function.

2) *Some typos and grammatical errors.*

Reply: Thank you very much. We have carefully revised the manuscript and remove all grammatical and typos using grammar checking tool.

3) *The computational complexity comparison of the proposed method with other methods should be provided. Authors may want to conduct a time complexity analysis of the proposed algorithms, or at least discuss the factors/reasons for causing the runtime difference in Section K.*

Reply: Thank you very much for the valuable comment. We have split the one plot into three plots to conclude the computational complexity of proposed algorithm and discuss reasons of their computational performance in the revised manuscript (**see Page: 10, Section V**).

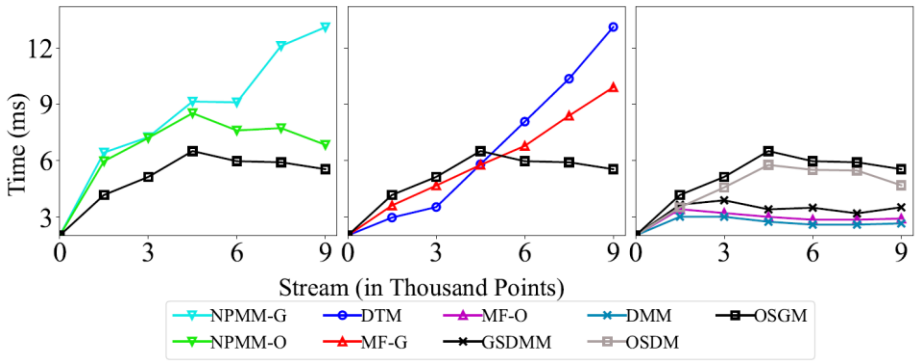


Figure 2 Running Time Comparison of Competing Algorithms

The time consumption of all algorithms on News-T dataset is shown in Figure, where we split algorithms into three plots to increase the difference visibility. The first plot reflects algorithms having higher execution time than OSGM, which are NPMM-G and NPMM-O. The reason is NPMM exploit pre-trained word embeddings to calculate posterior probability. The second plot shows moderate execution time algorithms which are MF-G and DTM. As we can clearly observe that both algorithms execution time abruptly increase as topics in stream increase, which directly effects the runtime of their iterative process for topic inference. The third plot shows algorithm having comparative a bit lower execution time than OSGM. The visible difference is due to extra computation required by OSGM to calculate term co-occurrence matrix for semantic similarity, whereas MF-O, GSDMM, and DMM lack this property.

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

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

***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 "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 paper, 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 existing approaches, our online model is not only free of determining optimal batch size, but also lends itself to handling large-scale data streams efficiently. It is also able to handle "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 data sets.**

***Index Terms*—Short text stream, Clustering, Graphical model, Topic Modeling, Topic evolution, Micro-clusters**

I. Introduction

A Short text stream is the continuous arrival of short-length documents over time. Nowadays, more and more short text data has been generated on social media and other platforms every day, e.g., 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, i.e., news recommendation [1], topic tracing [2], and 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.

This work is supported by the Fundamental Research Funds for the Central Universities (ZYGX2019Z014), National Natural Science Foundation of China (61976044), Fok Ying-Tong Education Foundation for Young Teachers in the Higher Education Institutions of China (161062), National key research and development program (2016YFB0502300).

J. Kumar, S.U. Din, Q. Yang, J. Shao are with School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, China (e-mail: jay@std.uestc.edu.cn, salahuddin@std.uestc.edu.cn, qinli.yang@uestc.edu.cn, junmshao@uestc.edu.cn)

J. Kumar, S.U. Din, Q. Yang, R. Kumar, J. Shao are also with with Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 313001, China.

Corresponding author: Junming Shao (junmshao@uestc.edu.cn)

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], [5], [6]. By transforming the traditional text clustering algorithms, similarity-based approaches were introduced such as HPStream [7], Feature Weighting K-Means (FW-KMeans) [8], DenStream [9], Spherical K-Means (SPKM) [10], COBWEB [11] and ConStream [12]. With a pre-specified 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 Dynamic Topic Model (DTM) [13], Temporal Text Mining (TTM) [14], Temporal Dirichlet process mixture model (TDPM) [15], Trend analysis model (TAM) [16], Temporal latent dirichlet allocation (TM-LDA) [17], DPMFP [18], and Gibbs sampling for dirichlet mixture model (GSDMM) [19], to mention a few. These approaches often require a pre-specified 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 evolving number of topics over streams is a non-trivial task.

Challenge 2: "Term Ambiguity" of Short-text Word Representation. Additionally, unlike long text documents, short text clustering further suffers from the lack of supportive term occurrence to capture semantics [20] due to the sparse word representation. For most existing short text clustering algorithms like Sumblr [21], Dynamic cluster topic model (DCT) [22], MStreamF [23] and SIF-AE [24], exploiting independent word representation in their cluster models tends to cause "term ambiguity" [20] (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 [25] [26].

Challenge 3: Online Evolving Topic Modeling. Another potentially problem of most existing approaches is the batch-wise processing. The batch-wise processing assumes that the instances are interchangeable within a batch[1] [3], [27]. Determining an optimal batch size is also a non-trivial task for different text streams. These algorithms also need to process each batch multiple times to infer the clusters. For online processing, NPMM [28] 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, pre-trained word

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

embeddings are thus not efficient in real-time streams [29].

To clearly describe the mentioned problems, Figure 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 with 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: (a) automatically detect the number of topics over time using the Polya Urn scheme, (b) capture semantics by embedding evolving term co-occurrence matrix, (c) find the topic related term-subspace in online way, and (d) track cluster term evolution using feature-level triangular time decay. The main contributions of this paper are highlighted as follows.

- **Evolving Topic Modeling with Online Term Distribution Exploring:** Instead of batch-wise 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 evolving topics in changing term-subspaces.
- **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.
- **Robustness:** OSGM allows merging highly similar clusters, which yields the number of topics near to actual ones automatically. Moreover, unlike traditional dirichlet based non-parametric models, OSGM is more robust to the concentration parameter (Dirichlet Process parameter, cf. Section III-B) and background term noise parameter (Multinomial distribution coefficient, cf. Section III-B).

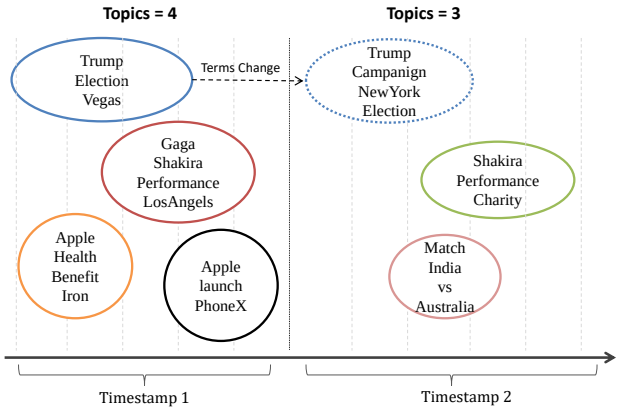


Fig. 1: Illustration of challenges of short text stream clustering with (i) evolving number of topics, (ii) term ambiguity, (iii) evolving term distribution problems. Here the size of the topic represent its active life-span in terms of unit time. For evolving topics, the number of topics is unknown 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 topic's subspace, the distribution of core terms changes.

II. Related Work

Existing clsutering on text streams can be generally divided into two categories: Model-based clustering and similarity-based clustering [4], [5]. Former clustering methods are mostly based on statistical models. Whereas, the latter can also be referred to as the threshold-based approaches.

A. Model-based Clustering

Latent Dirichlet Allocation (LDA) [30] 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 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 dynamic topic model (DTM) [13], dynamic mixture model (DMM) [31], online clustering of text streams (OCTS) [32], temporal LDA (T-LDA) [17], and streaming LDA (S-LDA) [33]. These models assume that each document contains rich content, and thus they are not suitable to deal with the short text streams. Later, Dirichlet multinomial mixture model-based dynamic clustering topic model (DCT) is proposed to deal with short text streams by assigning each document with a single topic [22]. However, these models need a high value of k (number of clusters) to generate clusters through dirichlet process. Later on, GSDMM [19] 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 [34], [35].

To automatically detect the number of clusters, Ahmed et al. [15] propose a temporal dirichlet process mixture model (TDMP). 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 [37] 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 TDMP and GSDPMM models do not examine the evolving topics by maintaining the active topics. Thereafter, MStreamF [23] 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 non-trivial task. Along with the disadvantage of fixed batch size, it also cannot deal with the term ambiguity problem. Recently, NPMM model [26] 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 needs to be pre-trained, hence cannot capture evolving semantics for generating quality clusters. Therefore, capturing semantics by removing term ambiguity is a non-trivial task while dealing with online clustering of text streams. Our proposed approach can solve term ambiguity issue in online way while dealing with evolving clusters.

Additionally, 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. [38], [32] propose term smoothing to reduce the dimensions in model-based approaches which helps to highlight core terms. NPMM [26] maintains a bit vector to represent 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 cluster feature set, instead, each cluster is represented by terms of its documents. Due to the multinomial assumption of dirichlet process, 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. [39]. The framework is composed of online and offline clustering modules. It is supposed to assign a much higher number of micro-clusters than the number of actual macro-clusters. Later on, many popular clustering algorithms are mapped to this general framework. Zhong et al. apply an adaptive spherical K-means clustering on streaming data [10]. The proposed approach creates unit-length vectors for all the micro-cluster centriod. However, scalability and sparsity issues still exist [28]. Later on, OSKM [10] 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. 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 [12] is exploited to deal with text data in stream. A *cluster droplet* is introduced as cluster feature, which represents only those terms related to local space. By using the decay mechanism, outliers are also detected by calculating the cosine similarity between micro-clusters. DenStream [9] is introduced by designing DBSCAN algorithm for streaming data. Unlike other algorithms, this approach does not require a static number of clusters. However, the minimum number of data points and radius are still required. 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

In this section, some preliminaries are first given, which include the notations and notions, followed by description of a Dirichlet process. Polya Urn Scheme and Chinese Restaurant Process are further explained, so that the statistical procedure of infinite generation of topics can be understood.

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 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 Dirichlet Process (DP) is a non-parametric 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 [40]. The scheme is summarized as follows.

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

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 exist in urn with probability of $\frac{n-1}{\alpha+n-1}$ or draw a new color with probability of $\frac{\alpha G_0}{\alpha+n-1}$. 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 process called the Chinese Restaurant Process [41].

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

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

The mixture weights $\theta = \{\theta_k\}_{k=1}^{\infty}$ can be formalized by $\theta \sim GEM(\gamma)$ [42]. We exploit Equation (2) for the generative process of the Dirichlet process multinomial mixture model (DPMM) [37] as follows :

$$z_d|\theta \sim \text{Mult}(\theta) \quad d = 1, \ldots, \infty$$
$$G_k|\beta \sim \text{Dir}(\beta) \quad k = 1, \ldots, \infty$$
$$d\,|z_d, \{G_k\}_{k=1}^{\infty} \sim p\left(d|G_{z_d}\right)$$

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:

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

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 document and 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 $\frac{n_k}{\alpha+n-1}$ for choosing existing topic and $\frac{\alpha}{\alpha+n-1}$ for choosing new topic. The term $p(d|G_z)$ represents the similarity between topic (table) and document (customer), multinomial distribution with dirichlet process is used for existing topic and for the new topic as well.

IV. Proposed Approach

In this section, we give detailed description of cluster feature set and then introduce our online semantic-enhanced graphical model (OSGM) for evolving short text stream clustering.

A. Cluster Feature Set

A micro-cluster in stream is represented by cluster feature set which contains different statistical values about the instances within a micro-cluster. Most similarity-based and model-based methods follow the vector space model (VSM) to represent the cluster feature space [43]. However, a topic needs to be represented in the term-changing subspaces. Here, an extended cluster feature (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 $\{m_z, n_z^w, cw_z, n_z, l_z, u_z, ta_z\}$. 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 cluster.

$$m_z = m_z + 1$$
$$n_z^w = n_z^w + N_d^w \quad \forall w \in d$$
$$cw_z = cw_z \cup cw_d$$
$$n_z = n_z + N_d$$
$$ta_z = ta_z \uplus ta_d \quad \forall w \in d$$

Here, cw_d is co-occurrence of each word with every word of same document, and N_d represents the number of total words in the document. $\uplus\ ta_d$ merges the arrival time of each document term (Example is provided in *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 micro-cluster 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

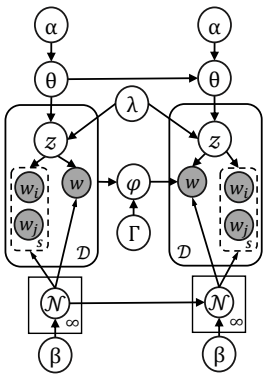


Fig. 2: The graphical model of OSGM. The shaded are observed and unshaded are latent entities.

and transform the MStreamF model from batch-wise to 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 term coefficient. The φ controls the change of topic related terms over time with Γ as 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 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: eliminate term ambiguity with calculating document-cluster similarity, handle evolving number of topics[2], determine the evolving topics in changing term-subspaces, and remove outdated topics. The pseudo code of our proposed approach is given in Algorithm 1.

1) Document-Cluster Similarity: Initially the first arriving document creates its own cluster. For 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 similarity between a document and an existing cluster, we transform Equation (4) and derive the probability of an arriving document d to choose an existing cluster z as:

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

The derivations of the equation is provided in *supplementary material*. The first term $\left(\frac{m_z}{D-1+\alpha D}\right)$ represents $p(G_z)$ of Equation (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 Equation (4). The first part captures similarity in singular term space (the middle term of the Equation (5)) and second part captures similarity in semantic term space (co-occurrence of terms) to solve term-ambiguity problem. The middle term is based on the multinomial distribution (see Equation (3)) with a pseudo weight of words β defines the homogeneity between a cluster and a document. Previously, 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 similar weight measure called inverse cluster frequency ICF_w to calculate term importance, defined as:

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

The denominator part of Equation (6) is the number of those clusters which contains the word w, and $|Z|$ is total number of active clusters. 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, suppose we have two clusters $C1$ and $C2$.

$$n_{C1}^w = \{w_1 : 3, w_2 : 4, w_1 : 3, w_3 : 1, w_4 : 2\}$$
$$n_{C2}^w = \{w_1 : 3, w_2 : 4, w_1 : 3, w_5 : 1, w_6 : 2\}$$

$$cw_{C1} = \begin{bmatrix} w_1 & w_2 & w_3 & w_4 & \\ - & 0.5 & 0 & 0 & w_1 \\ 0.5 & - & 0.25 & 0 & w_2 \\ 0 & 0.75 & - & 0.5 & w_3 \\ 0 & 0 & 0.5 & - & w_4 \end{bmatrix}$$

$$cw_{C2} = \begin{bmatrix} w_1 & w_2 & w_5 & w_6 & \\ - & 0 & 0 & 0.5 & w_1 \\ 0 & - & 0.75 & 0 & w_2 \\ 0 & 0.25 & - & 0.5 & w_5 \\ 0.5 & 0 & 0.5 & - & w_6 \end{bmatrix}$$

We assume $d' = \{w_1 : 1, w_2 : 1, w_7 : 1\}$ as an arriving document , and calculate the similarity of d' with both clusters. As we can observe that the w_1 and w_2 are common between both clusters. If we do not consider co-occurrence then the probability of both clusters are same, however, w_1 and w_2 co-occurred in $C1$. Therefore, the third part $\left(1 + \sum_{w_i \in d \wedge w_j \in d} cw_{i,j}\right)$ in Equation (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.

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

Here, $n_{d'}^{w_i}$ is the frequency count of word w_i in the document

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

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

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 *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 cluster feature 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 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 Equation (4) and derive probability as follows.

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

Here, $\bar{V}_z$ is the average vocabulary size of active clusters. The 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 Algorithm 1).

3) Online Evolving Topic Extraction in Changing term-subspaces: The previous study [44] argues that a topic related text document has few core terms to represent a topic. Whereas, for some topics, distribution of core terms also changes over time. To highlight core terms and term subspace change in online way, we first maintain arriving timestamps of each term. We then calculate age of the cluster by adopting triangular number of its size (see Line 5 Algorithm 2). Afterwards, we compare arrival timestamps of each term with age of cluster to reduce noisy terms. The triangular time is originally employed to fade out outdated micro-clusters [45], defined as,

$$\text{Triangular Number } \Delta f(T) = \left(\left(T^2+T\right)/2\right). \quad (9)$$

Here, T is the timestamp number. The recency score of a term is measured by ratio of the sum of arrivals and age of cluster (see Line 8 Algorithm 2). For example, cluster z contains 9 documents then the age of cluster z will be:

$$Age_z = \Delta f(9) - \Delta f(1) = ((9^2+9)/2) - ((1^2+1)/2)$$

Let us suppose $D1$, $D2$ 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 is stored as:

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

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

Algorithm 1: OSGM

Input: $S_t : \{d_t\}_{t=1}^{\infty}$, α : concentration parameter, β : pseudo weight of term in cluster, λ : decay factor, Γ : feature decay threshold

Output: Cluster assignments z_d

```
1  initialize M = φ                          // empty model
2  t_c = 0                                   // model timestamp
3  while d_t in S_t                          // start streaming
4  do
5      t_c = t_c + 1
6      M = updateActiveClusters(λ, M) // Alg. 3
7      M = updateTermsSubspace(Γ, M) // Alg. 2
8      foreach z_i ∈ M do
9          if V_d ∩ V_z ≠ φ then
10             P_{Z_i} = p(z_d = z_i, d_t) using Equation (5)
11         end
12     end
13     i = arg max_i (P_{Z_i})
14     P_{Z_n} = p(z_d = z_new, d_t) using Equation (8)
15     if P_{Z_i} < P_{Z_n} then
           // create new cluster
16         m_{z_n} = 1, n^w_{z_n} = N^w_{d_t}
17         cw_{z_n} = cw_{d_t} , n_{z_n} = N_{d_t}
18         l_{z_n} = 1, u_{z_n} = t_c, ta_{z_n} = ta_{d_t}
19         M = M ∪ z_n
20     else
21         update z_i using Definition 1
22         l_{z_i} = 1, u_{z_i} = t_c
23     end
24 end
```

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

$$recency_{\text{gaga}} = ((1+2+8)\times 100)/Age_z$$

if the $recency$ is less than our defined threshold Γ then we consider that this term cannot represent current concept of cluster. In other words if a term is frequently observed in cluster with the passage of cluster time-span then the term is useful to represent the current concept of terms (see Line 9 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 whole 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 real time environment, each topic may have 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 active for two days. Many existing approaches often delete old clusters using some for-

Algorithm 2: Update Cluster Term-subspace

```
/* Update weight of terms in each
   active cluster                          */
Function updateTermsSubspace(Γ, M)
    S_z = Δf(1) using Equation (9)
    foreach z ∈ M do
        E_z = Δf(m_z) using Equation (9)
        Age_z = E_z − S_z + 1
        foreach (w, ta_w) ∈ ta_z do
            // sum arrival timestamps of w
            sumta_w = Σ_{t∈ta_w} t
            recency_w = (sumta_w × 100)/Age_z
            if recency_w < Γ then
                remove(w, z)
            end
        end
    end
    return M
```

Algorithm 3: Update Active Clusters

```
/* Update weight of all active
   clusters                                */
Function updateActiveClusters(λ, M)
    Old = φ
    foreach z ∈ M do
        calculate l_z using Equation (10)
        if l_z ≈ 0 then
            Old = Old ∪ z
        end
    end
    foreach z_i ∈ Old do
        foreach z_act ∈ (M − Old) do
            if V_{z_act} ∩ V_{z_i} ≠ φ then
                P_{Z_i} = p(z_d = z_act, z_i) using Equation
                (5)
            end
        end
        max = arg max_i (P_{Z_i})
        P_{Z_n} = p(z_d = z_new, z_i) using Equation (8)
        if P_{Z_max} < P_{Z_n} then
            M = M − z_i
        else
            merge(z_i, Z_max) using Definition 1
        end
    end
return M
End Function
```

getting mechanisms (e.g., decay rate), some deletes instances of old batches [10][12][28][23]. The life-span of each document depends on the active time period of a topic, therefore, in contrast to deleting old batches we adopt 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,

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

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 Algorithm 1). Importance score of a cluster decreases over time if it is not receiving any document. If the score of a cluster approximately reaches to zero (see Line 7 Algorithm 3), then the cluster is processed to be removed from the model, i.e., 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 micro-clusters 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 active cluster by calculating the probability using Equation (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.

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 other side, OSGM calculates the probability of arriving document with almost each active cluster (see Line 8 of Algorithm 1). Additionally, 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 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 [37], [22], [46], [23], [47], [26] to evaluate short text clustering models. In the preprocessing step, we removed stop words, converted all text into lowercase, and stemming. The description of the datasets is as follows.

(1) **News (Ns):** This dataset is collected by [48]. It contains 11,109 title 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 period and then disappear. The next thing to observe that in

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

TABLE II: Statistics of Each dataset where Ds represent name of dataset, $|D|$ represents total number of documents, $|V|$ is number of unique terms, $|T|$ is total number of topics, and $\mathcal{L}$ is 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

many situations the distribution of topic related terms changes. We constructed synthetic dataset by injecting two mentioned conditions in News dataset as,

1) we sorted documents by topics. After sorting, we then divide the dataset into sixteen equal chunks and shuffled them.
2) we analyzed 5 main topics whose term distribution changed in sub-topics over time. To depict change in term distribution, we combined sub-topics and placed them at continuous position.

The Fig. 3a shows the instance position of 5 topics in stream. Here, black point instances show the initial distribution of terms of a new topic, whereas, the green points depicts instances of the term distributional change (concept drift) of same topic. An example from this dataset is given in Fig. 4.

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

The statistics of the datasets are reported in Table II. Here, both datasets contain less than 10 average document length ($\mathcal{L}$) which assure datasets fit for short text analysis. Specifically, the News dataset contains lower average document length, compare to 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 i) OSGM and ii) OSGM-ES with baselines. The former approach includes semantic smoothing of inverse

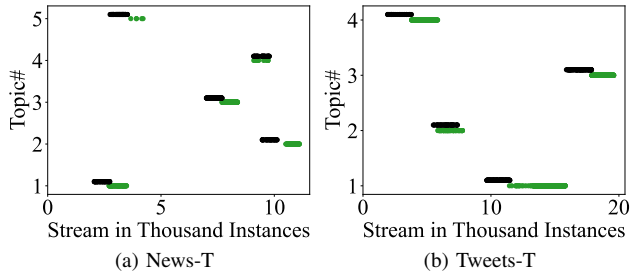


Fig. 3: In News-T and Tweets-T dataset, there are five and four main topics, respectively. Each topic is divided into two sub-topics and the plot shows the instances of these sub-topics in which change in term distribution occurs.

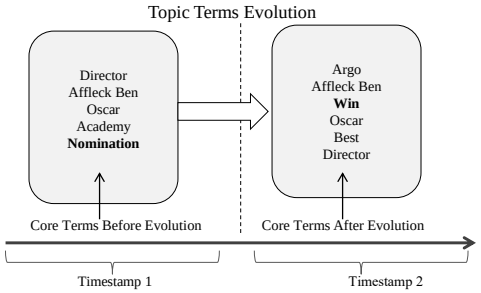


Fig. 4: The changes of core terms for a topic from News.

clustering frequency, whereas latter approach excludes it. We have selected five state-of-the-art representative algorithms for text stream clustering to comparsion.

(1) **DMM** [48] is an initial model based on multinomial dirichlet modeling to simulate CRP.
(2) **OSDM** [49] is the most recent model to deal infinite number of topics in online way.
(3) **DTM** [13] is an extension of Latent Dirichlet Allocation which traces the evolution of hidden topics from corpus over time. It was designed to deal with the sequential documents.
(4) **GSDMM** [48] 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 iterative Gibbs sampling procedure.
(5) **MStreamF** [23] 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 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** [26] is the latest model to deal with the infinite number of latent topics in short texts while exploiting pre-trained Glove word-embeddings to capture semantics. Authors proposed two variations of their algorithm, one with iterative process referred 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). The purity of cluster $z \in Z$ is defined by,

$$pur(z) = \frac{1}{|z|} \max(n_{c,z} | c \in C) \tag{11}$$

Here, $|z|$ is the size of cluster, C represent all classes, and $n_{c,z}$ reflects the number of instances in z related to class c. Homogeneity measures that each cluster should have only members of a single class, defined as:

$$Homo = 1 - \mathrm{H}(C|Z) \times H(C). \tag{12}$$

Here, $H(C|Z)$ is the conditional entropy and $H(C)$ represent the coverage of classes, defined as,

$$H(C|Z) = \sum_{z,c} \frac{n_{c,z}}{N} \cdot \log\left(\frac{n_{c,z}}{\sum_c n_{c,z}}\right)$$

$$H(C) = \sum_{c} \frac{\sum_z n_{c,z}}{N} \cdot \log\left(\frac{\sum_z n_{c,z}}{N}\right).$$

Here, the total number of documents is represented by N. The V-measure [50] calculates how successfully the criteria of completeness and homogeneity are satisfied, defined as,

$$V\text{-}M = \frac{2 \cdot (Homo \times Co)}{(Homo + Co)}. \tag{13}$$

Here, $\mathrm{Co} = 1 - \mathrm{H(Z|C)} \times \mathrm{H(Z)}$ and each term is defined as,

$$H(Z|C) = \sum_{c,z} \frac{n_{c,z}}{N} \log\left(\frac{n_{c,z}}{\sum_z n_{c,z}}\right)$$

$$H(Z) = \sum_{z} \frac{\sum_c n_{c,z}}{N} \cdot \log\left(\frac{\sum_c n_{c,z}}{N}\right).$$

The NMI measure calculates overall clustering quality, defined as,

$$NMI = \frac{\sum_{c,z} n_{c,z} \log\left(\frac{N \cdot n_{c,z}}{n_c \cdot n_z}\right)}{\sqrt{\left(\sum_c n_c \log\frac{n_c}{N}\right)\left(\sum_z n_z \log\frac{n_z}{N}\right)}} \tag{14}$$

Here, n_z is number of documents in z and n_c is number of documents of class c. We utilized sklearn[6] API to compute the measures on overall clustering results [48].

D. Parameter Setting

We try to find the optimal parameter values for best results for most algorithms with grid search. Finally, we obtain best results on $\alpha = 0.01$ for DTM, $\alpha = 0.3$ and $\beta = 0.3$ for DMM. The DTM and DMM need fixed number of cluster 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 [23], we set the number of iterations to 10 and $saved\text{-}batches = 2$ for MStreamF-G. As defined in [26], 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, repectively. We set $\Gamma = 10$ and $\lambda = 1e^{-6}$ for both variants of 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.

[6]http://scikit-learn.org

TABLE III: The performance of different algorithms on four datasets in terms of different measures including Normalized Mutual Information (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 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 whole 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 *supplementary materials* due to space limitation.

F. Runtime Analysis

To compare the running time of competing algorithms, we performed all the experiments on a system with core i5-3470 and 8GB memory. The time consumption of all algorithms on 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 time than OSGM, which are NPMM-G and NPMM-O. The reason is NPMM exploit pre-trained word embeddings to calculate posterior probability. The second plot shows moderate execution time algorithms which are MF-G and DTM. As we can clearly observe that both algorithms execution time abruptly increase as topics in stream increase, which directly effects the runtime of their iterative process for topic inference. The third plot shows algorithm having comparative a bit lower execution time than OSGM. The visible difference is due to extra computation required by OSGM to calculate 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 with most existing algorithms.

G. Parameter Sensitivity Analysis

We perform sensitivity analysis for OSGM with respect to four input parameters: α, β, λ, and Γ on the News dataset. Fig. 6a shows the effect of α which ranges from $1e^{-3}$ to $1e^{-1}$. It can be observed that the performance of OSGM is quite stable over a large range of α values. The parameter α also contributes to infer the number of clusters, that is why the performance is stable. The multinomial distribution β parameter is responsible while calculating the term distribution similarity. Fig. 6b 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 towards 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 to zero. Fig. 6c depicts the performance of OSGM on different decay factors ranging from $9e^{-7}$ to $9e{-}5$. It can be observed that the

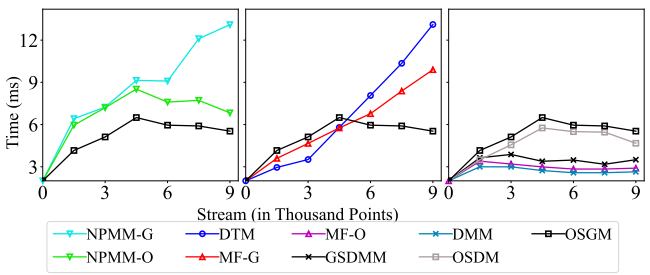


Fig. 5: The runtime in milliseconds (ms) of different text stream clustering algorithms.

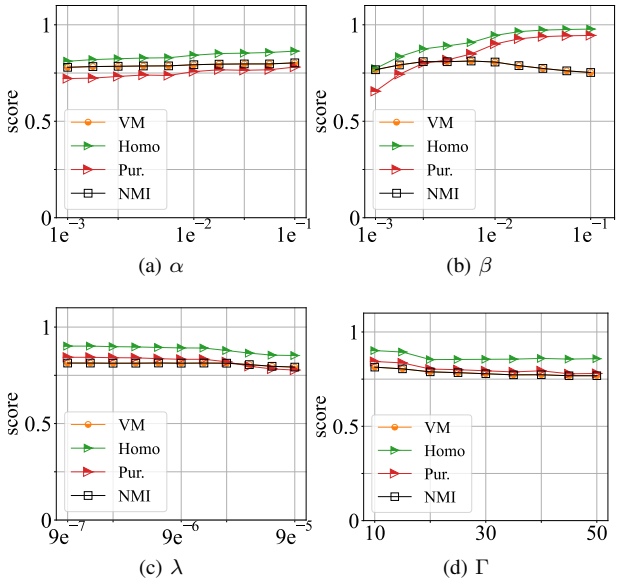


Fig. 6: Parameter sensitivity analysis.

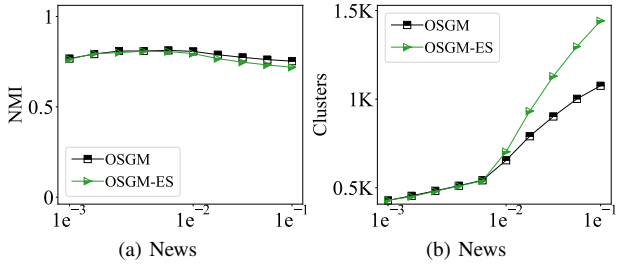
behavior of a given evaluation measure is stable and does not show high variation. The triangular time threshold Γ is responsible to capturing evolving features and extracting core terms. Fig. 6d shows the sensitivity of a defined threshold ranging from 10 to 50. Interestingly, the performance is stable 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 towards highly fine-grained clustering.

H. Semantic Smoothing Analysis

For the 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. 7a 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 don't use semantic smoothing. Likewise, Fig. 7b shows semantic smoothing leads towards a high NMI score by following coarse-grain clustering.

I. Topic Estimation Analysis

The OSGM automatically creates clusters based on calculated probability for evolving number of topics, thus it does not need number of topics as input. Fig. 8 shows the active number of clusters and actual number of topics over time comparative to different algorithms including NPMM-O, OSDM, MStreamF-O (MF-O), and MStreamF-G. To interprets the given plots, suppose, our model has $|D|$ number of active documents at time-stamp t, and according to ground truth



(a) News (b) News

Fig. 7: β sensitivity of OSGM and OSGM-ES on real dataset in terms of NMI and cluster creation.

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 the "Actual" represents the $|\mathcal{L}|$ and "OSGM" reflects the $|M|$ of the model over time. By observing the model cluster over time we can conclude that OSGM starts converging towards actual topics as clusters start merging with the passage of timestamps. However, OSDM and MF-O does not infer clusters in any way, that is the reason increase in clusters is observed continuously. In contrast, clusters of MF-G seem much closer due to batch-wise iterative process. Unlike other algorithms, NPMM uses pre-trained word embeddings (which already contains a closer distribution of words), therefore the number of clusters are much closer

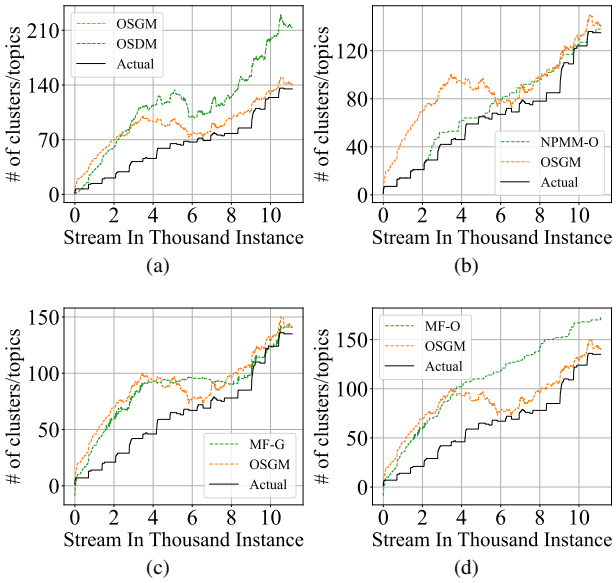


Fig. 8: Predicted vs actual number of topics.

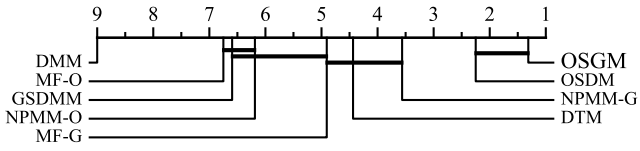


Fig. 9: Nemenyi Test on all data sets (real + synthetic) for OSGM.

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

J. Statistical Tests

The series of previous state-of-the-art algorithms follows the similar non-parametric statistical model to assess the statistical significance between comparing approaches. As reported in [51], we perform Friedman rank test with 95% confidence level. We define the null hypothesis as no statistical difference among competing methods. If the null hypothesis is rejected, we further use 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 get 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 with OSGM, which can deal with evolving topics but cannot deal with evolving term-subspace.

VI. Conclusion

In this paper, we propose a new online semantic enhanced graphical model for evolving short text stream clustering. Compared to existing approaches, OSGM does not require to specify the batch size, the dynamic number evolving clusters, and can reduce cluster features 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 investigate the importance of semantic smoothing in the proposed model. A deep conducted empirical study on synthetic and real-world datasets further demonstrates 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," *Information Sciences*, vol. 254, pp. 1–18, 2014.
[2] P. Li, L. He, H. Wang, X. Hu, Y. Zhang, L. Li, and X. Wu, "Learning from Short Text Streams with Topic Drifts," *IEEE Transactions on Cybernetics*, vol. 48, no. 9, pp. 2697–2711, 2018.
[3] K.-Y. Chen, L. Luesukprasert, and T. C. Seng-cho, "Hot topic extraction based on timeline analysis and multidimensional sentence modeling," *IEEE transactions on knowledge and data engineering*, vol. 19, no. 8, pp. 1016–1025, 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 Computing 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," *Knowledge and Information Systems*, 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, 2015.

[7] C. C. Aggarwal, J. HAN, J. WANG, and P. S. Yu, "A Framework for Projected Clustering of High Dimensional Data Streams," *International conference on Very large data bases*, pp. 852–863, 2004.

[8] L. Jing, M. K. Ng, J. Xu, and J. Z. Huang, "Subspace Clustering of Text Documents with Feature Weighting K-Means Algorithm," *Advances in Knowledge Discovery and Data Mining*, vol. 3518, pp. 802–812, 2005.

[9] F. Cao, M. Estert, W. Qian, A. Zhou, M. Ester, W. Qian, and A. Zhou, "Density-Based Clustering over an Evolving Data Stream with Noise," *Proceedings of SIAM International Conference on Data Mining*, pp. 328–339, 2006.

[10] S. Zhong, "Efficient streaming text clustering," *Neural Networks*, vol. 18, no. 5-6, pp. 790–798, 2005.

[11] N. Sahoo, J. Callan, R. Krishnan, G. Duncan, and R. Padman, "Incremental hierarchical clustering of text documents," in *International Conference on Information and Knowledge Management*, 2006, p. 357.

[12] C. C. Aggarwal and P. S. Yu, "On Clustering Massive Text and Categorical Data Streams," *Knowledge and Information Systems*, vol. 24, no. 2, pp. 171–196, 2010.

[13] D. M. Blei and J. D. Lafferty, "Dynamic topic models," *ACM International Conference Proceeding Series*, vol. 148, pp. 113–120, 2006.

[14] Q. Mei and C. X. Zhai, "Discovering evolutionary theme patterns from text - An exploration of Temporal Text Mining," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 198–207, 2005.

[15] A. Ahmed and E. P. Xing, "Dynamic Non-Parametric Mixture Models and The Recurrent Chinese Restaurant Process: with Applications to Evolutionary Clustering," in *Proceedings of SIAM International Conference on Data Mining*, 2008, pp. 219–230.

[16] N. Kawamae, "Trend analysis model: Trend consists of temporal words, topics, and timestamps," *Proceedings of the 4th ACM International Conference on Web Search and Data Mining, WSDM 2011*, pp. 317–326, 2011.

[17] Y. Wang, E. Agichtein, and M. Benzi, "TM-LDA: Efficient Online Modeling of Latent Topic Transitions in Social Media," in *International conference on Knowledge discovery and data mining*. ACM, 2012, pp. 123–131.

[18] R. Huang, G. Yu, Z. Wang, J. Zhang, and L. Shi, "Dirichlet process mixture model for document clustering with feature partition," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 8, pp. 1748–1759, 2013.

[19] J. Yin and J. Wang, "A Text Clustering Algorithm Using an Online Clustering Scheme for Initialization," in *ACM International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1995–2004.

[20] H. Gong, T. Sakakini, S. Bhat, and J. Xiong, "Document similarity for texts of varying lengths via hidden topics," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, vol. 1, 2018, pp. 2341–2351.

[21] L. Shou, Z. Wang, K. Chen, and G. Chen, "Sumblr Continuous Summarization of Evolving Tweet Streams," in *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2013, pp. 533–542.

[22] S. Liang, E. Yilmaz, and E. Kanoulas, "Dynamic Clustering of Streaming Short Documents," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pp. 995–1004, 2016.

[23] J. Yin, D. Chao, Z. Liu, W. Zhang, X. Yu, and J. Wang, "Model-based Clustering of Short Text Streams," in *ACM International Conference on Knowledge Discovery and Data Mining*, 2018, pp. 2634–2642.

[24] A. Hadifar, L. Sterckx, T. Demeester, and C. Develder, "A Self-Training Approach for Short Text Clustering," in *Proceedings of the 4th Workshop on Representation Learning for NLP (ACL)*, 2019, pp. 194–199.

[25] Z. Haj-Yahia, A. Sieg, and L. A. Deleris, "Towards Unsupervised Text Classification Leveraging Experts and Word Embeddings." Association for Computational Linguistics, 2019, pp. 371–379.

[26] J. Chen, Z. Gong, and W. Liu, "A nonparametric model for online topic discovery with word embeddings," *Information Sciences*, vol. 504, pp. 32–47, 2019.

[27] 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, 2019.

[28] M. K. Islam, M. M. Ahmed, and K. Z. Zamli, "A buffer-based online clustering for evolving data stream," *Information Sciences*, vol. 489, pp. 113–135, 2019.

[29] X. Cheng, X. Yan, Y. Lan, and J. Guo, "BTM: Topic modeling over short texts," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 12, pp. 2928–2941, 2014.

[30] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[31] X. Wei, J. Sun, and X. Wang, "Dynamic mixture models for multiple time-series," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, M. M. Veloso, Ed., 2007, pp. 2909–2914.

[32] Y. B. Liu, J. R. Cai, J. Yin, and A. W. C. Fu, "Clustering text data streams," *Journal of Computer Science and Technology*, vol. 23, no. 1, pp. 112–128, 2008.

[33] H. Amoualian, M. Clausel, E. Gaussier, M.-R. Amini, M. Clausel, M.-R. Amini, É. Gaussier, and M.-R. Amini, "Streaming-LDA: A Copula-based Approach to Modeling Topic Dependencies in Document Streams," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 695–704.

[34] S. U. Din, J. Shao, J. Kumar, W. Ali, J. Liu, and Y. Ye, "Online Reliable Semi-supervised Learning on Evolving Data Streams," *Information Sciences*, vol. 507, 2020.

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

[36] J. Sui, Z. Liu, L. Liu, A. Jung, and X. Li, "Dynamic sparse subspace clustering for evolving high-dimensional data streams," *IEEE Transactions on Cybernetics*, 2020.

[37] Y. Jianhua and J. Wang, "A model-based approach for text clustering with outlier detection," in *32nd IEEE International Conference on Data Engineering*. IEEE Computer Society, 2016, pp. 625–636.

[38] Y. Liu, J. Cai, J. Yin, and A. W. Fu, "Clustering massive text data streams by semantic smoothing model," vol. 4632, pp. 389–400, 2007.

[39] C. C. Aggarwal, J. Han, J. Wang, P. S. Yu, J. W. Jiawei Han, P. S. Yu, J. Han, J. Wang, and P. S. Yu, "A Framework for Clustering Evolving Data Streams," in *International conference on Very large data bases*, 2003, pp. 81–92.

[40] D. Blackwell, J. B. MacQueen, Others, and D. R. Brillinger, "Ferguson distributions via Pólya urn schemes," *The annals of statistics*, vol. 1, no. 2, pp. 353–355, 1973.

[41] T. S. Ferguson and Thomas S Ferguson, "A Bayesian analysis of some nonparametric problems," *The annals of statistics*, vol. 1, no. 2, pp. 209–230, 1973.

[42] R. M. Neal, "Markov chain sampling methods for Dirichlet process mixture models," *Journal of computational and graphical statistics*, vol. 9, no. 2, pp. 249–265, 2000.

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

[44] G. Huang, J. He, Y. Zhang, W. Zhou, H. Liu, P. Zhang, Z. Ding, Y. You, and J. Cao, "Mining streams of short text for analysis of world-wide event evolutions," *World Wide Web*, vol. 18, no. 5, pp. 1201–1217, 2015.

[45] M. Tennant, F. Stahl, O. Rana, and J. B. Gomes, "Scalable real-time classification of data streams with concept drift," *Future Generation Computer Systems*, vol. 75, pp. 187–199, 2017.

[46] J. Qiang, Y. Li, Y. Yuan, and X. Wu, "Short text clustering based on Pitman-Yor process mixture model," *Applied Intelligence*, vol. 48, no. 7, pp. 1802–1812, 2018.

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

[48] J. Yin and J. Wang, "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*. ACM, 2014, pp. 233–242.

[49] J. Kumar, J. Shao, S. Uddin, and W. Ali, "An online semantic-enhanced dirichlet model for short text stream clustering," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Association for Computational Linguistics, 2020, pp. 766–776.

[50] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*, J. Eisner, Ed. ACL, 2007, pp. 410–420.

[51] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

An Online Semantic-enhanced Graphical Model for Evolving Short Text Stream Clustering (Supplementary Material)

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

I. Examples

This section discusses some example related to cluster feature set (CF) to provide readers more clear insight about given formulation in the paper.

A. Co-occurrence Matrix Example

A co-occurrence matrix is a property of cluster feature set which stores the relationship between terms of a single document, Suppose, we have a document $d = \{w_1 : 3, w_2 : 2, w_3 : 3\}$, we calculate each entry of matrix as,

$$cw_{w_1,w_2} = \frac{3}{3+2} = 0.6$$

$$cw_{w_2,w_1} = \frac{2}{2+3} = 0.4$$

$$cw_{w_1,w_3} = \frac{3}{3+3} = 0.5$$

$$cw_{w_3,w_1} = \frac{3}{3+3} = 0.5$$

$$cw_{w_2,w_3} = \frac{2}{2+3} = 0.4$$

$$cw_{w_2,w_3} = \frac{3}{3+2} = 0.6$$

B. Addable Property of CF

In our model, a CF set is defined as a 7-tuple $\{m_z, n_z^w, cw_z, n_z, l_z, u_z, ta_z\}$. Suppose, we have cluster $z = \{d_1, d_2\}$ and its CF,

$$d_1 = \{w_1 : 3, w_2 : 2\}$$
$$d_2 = \{w_1 : 1, w_3 : 1, w_4 : 1\}$$

$$m_z = 2$$
$$n_z^w = \{w_1 : 4, w_2 : 2, w_3 : 1, w_4 : 1\}$$
$$n_z = 10$$

$$cw_z = \begin{bmatrix} w_1 & w_2 & w_3 & w_4 & \\ - & 0.4 & 0.5 & 0.5 & w_1 \\ 0.6 & - & 0.0 & 0 & w_2 \\ 0.5 & 0.0 & - & 0.5 & w_3 \\ 0.5 & 0 & 0.5 & - & w_4 \end{bmatrix}$$

$$ta_{w_1} = \{timestamp : 1, timestamp : 2\}$$
$$ta_{w_2} = \{timestamp : 1\}$$
$$ta_{w_3} = \{timestamp : 2\}$$
$$ta_{w_4} = \{timestamp : 2\}$$
$$ta_z = \{ta_{w_1}, ta_{w_2}, ta_{w_3}, ta_{w_4}\}$$

and suppose we have a new document d_i to add in cluster $d_i = \{w_1 : 1, w_2 : 1\}$, then the CF of z will be,

$$m_z = \mathbf{3}$$
$$m_z = \mathbf{12}$$
$$n_z^w = \{w_1 : \mathbf{4}, w_2 : \mathbf{5}, w_1 : 3, w_3 : 1, w_4 : 2\}$$
$$ta_{w_1} = \{timestamp : 1, timestamp : 2, \textbf{timestamp:3}\}$$
$$ta_{w_2} = \{timestamp : 1, \textbf{timestamp:3}\}$$

As we can observe that a new timestamp is added, which is the size of cluster. In other words, the life-span of terms is measure over the scale of cluster size. The Another point to highlight, that new document contains w_1 and w_2, therefore, the new score will be updated by adding document score of word to word co-occurrence, which will be

$$cw_z = \begin{bmatrix} w_1 & w_2 & w_3 & w_4 & \\ - & \mathbf{0.9} & 0.5 & 0.5 & w_1 \\ \mathbf{1.1} & - & 0.0 & 0 & w_2 \\ 0.5 & 0.0 & - & 0.5 & w_3 \\ 0.5 & 0 & 0.5 & - & w_4 \end{bmatrix}.$$

II. Probability Derivation

This section provides a bit detailed discussion about derived probability of calculating the similarity between generated clusters and arriving document.

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 document and topic in terms of word distribution.

$$p(z_d|G_z) = p(G_z).p(d|G_z) \tag{1}$$

Here, $p(G_z)$ defines the probability of topic popularity (in CRP it is table popularity), which is $\frac{n_k}{\alpha+n-1}$ for choosing existing

This work is supported by the Fundamental Research Funds for the Central Universities (ZYGX2019Z014), National Natural Science Foundation of China (61976044), Fok Ying-Tong Education Foundation for Young Teachers in the Higher Education Institutions of China (161062), National key research and development program (2016YFB0502300).

J. Kumar, S.U. Din, Q. Yang, J. Shao are with School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, China (e-mail: jay@std.uestc.edu.cn, salahuddin@std.uestc.edu.cn, qinli.yang@uestc.edu.cn, junmshao@uestc.edu.cn)

J. Kumar, S.U. Din, Q. Yang, R. Kumar, J. Shao are also with with Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 313001, China.

Corresponding author: Junming Shao (junmshao@uestc.edu.cn)

topic and $\frac{\alpha}{\alpha+n-1}$ for choosing new topic. The term $p(d|G_z)$ represents the similarity between topic (table) and document (customer), multinomial distribution with dirichlet process is used for existing topic and for the new topic as well. By deriving the equation for the probability of topic popularity we define

$$p(G_z) = \left(\frac{m_z}{D-1+\alpha D}\right).$$

Here, m_z is the number of documents in a particular cluster (customers on existing table in CRP) and D reflects the number of active documents. For capturing new distribution, as controlled by the α concentration parameter, we define

$$p(G_{New}) = \left(\frac{\alpha D}{D-1+\alpha D}\right),$$

where αD defines pseudo portion of active documents for new cluster (topic). To simulate the homogeneity in generative process we derive probability of a document generated by a particular topic as:

$$p(d|G_z) = p(w_z|w_d).p(w_z^i w_z^j | w_d^i w_d^j) \text{ where } i \neq j$$

Here, we assumes that topic related terms are generated by mixture of independent terms and related-terms generation (for capturing term semantics). The first part of equation reflect the probability of independent terms and the predecessor part for term relativeness. We derive the probability for independent terms as,

$$p(w_z|w_d) = \left(\frac{\prod_{w\in d}\prod_{j=1}^{N_d^w} p(w_z^j)+\beta+j}{\prod_{i=1}^{N_d} p(w_z^j)+(\beta V_{z\cup d})+i}\right) \quad (2)$$

The nominator part calculate the probability of words contain by cluster and document $w \in d$, The denominator part normalizes the equation by multiplying probability of all words of cluster which uses the global term space $V_{z\cup d}$ by taking union of unique term of both document and cluster. The correlation of terms in topic generation is derived as,

$$p(w_z^i w_z^j | w_d^i w_d^j) = \sum_{(w_i,w_j)\in \mathcal{V}_{d\cap z}} \frac{p(w_i w_j|z)}{p(w_i|z)+p(w_j|z)}$$

Here, $p(w_i w_j|z)$ is probability of two words co-occurring. As we know that $p(w_z|w_d) \leq 1$ therefore we combine them by adding 1 in $p(w_i w_j|z)$ as,

$$p(d|G_z) = p(w_z|w_d).\left(1+\sum_{(w_i,w_j)\in \mathcal{V}_{d\cap z}} \frac{p(w_i w_j|z)}{p(w_i|z)+p(w_j|z)}\right)$$

III. Clustering Quality and Term Evolution Analysis

This section provides additional experimental analysis of OSGM related to change in core terms of a topic.

To prove the superiority of OSGM, we conduct a deep analysis of clustering quality compared with word-embedding approaches. Fig. 2 shows the frequency of some terms found in datasets that are ignored by previous word-embedding based approaches when clustering the documents. In contrast, the proposed approach does not rely on any external embedding,

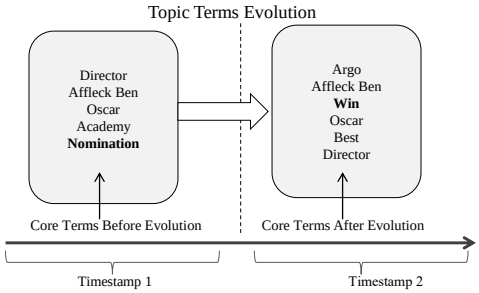


Fig. 1: The changes of core terms for a topic from News.

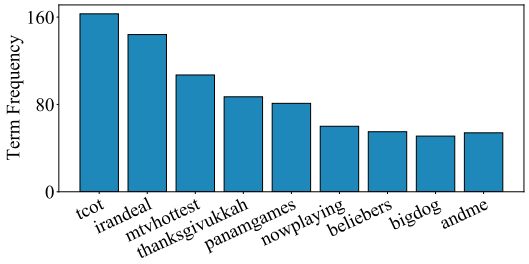
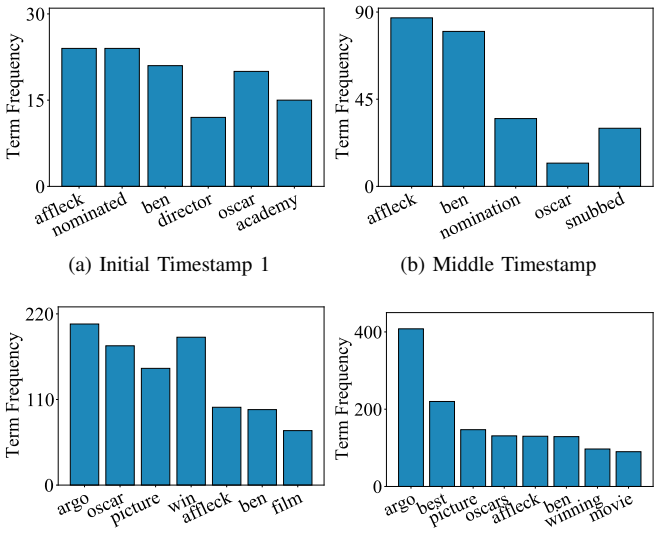


Fig. 2: Term occurrence of few topic related terms ignored by word-embedding based approaches, which is successfully captured by OSGM.



(a) Initial Timestamp 1 (b) Middle Timestamp

(c) Term-subspace Evolution Timestamp (d) Last Timestamp

Fig. 3: Term occurrence over different timestamps captured by OSGM. It is clearly observed that it successfully learns evolving active terms for a given topic.

instead, it learns from the current stream of documents. Additionally, we choose a topic as a case study to exploit the cluster term-subspace evolution. Fig. 1 shows the actual evolution of terms over time. We track the distribution of core-terms captured by OSGM related to the same topic at four different timestamps, as depicted in Fig. 3. It is observed that OSGM successfully traces the term evolution and can extract the core terms.