Received March 26, 2021, accepted May 3, 2021, date of publication May 6, 2021, date of current version May 18, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3078096

Lifelong Learning Augmented Short Text Stream Clustering Method

JIPENG QIANG, WANYIN XU, YUN LI, YUNHAO YUAN, AND YI ZHU
Department of Computer Science, Yangzhou University, Yangzhou 225127, China

Corresponding author: Yun Li (liyun@yzu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 62076217, Grant 61906060, and Grant 61703362; and in part by the Natural Science Foundation of Jiangsu Province of China under Grant BK20170513.

ABSTRACT Depending on the scanning mode, existing short text stream clustering methods can be divided into the following two kinds of methods: one-pass-based and batch-based. The one-pass-based method handles each text only one time, but cannot deal with the sparseness problem very well. The batch-based method obtains better results by allowing multiple iterations of each batch, but the efficiency is relatively low. To overcome these problems, this paper presents Lifelong learning Augmented Short Text stream clustering method (LAST), which incorporates the episodic memory module and sparse experience replay module of lifelong learning into the clustering process. Specifically, LAST processes each text one time, but at a certain interval it randomly samples some previously seen texts of the episodic memory to update cluster features by performing sparse experience replay. Empirical studies on two public datasets demonstrate that the performance of the LAST-based method is on a par with the batch-based method, and runs close to the speed of the one-pass-based method.

INDEX TERMS Short text stream, text clustering, sparseness, lifelong learning.

I. INTRODUCTION

Short texts are prevalent on the Web, including on traditional websites, e.g., news titles and search snippets, and emerging social media, e.g., microblogs and tweets. In recent years, these data have swept the world at an alarming rate, and have produced large quantities of data streams, also called short text streams. Short text stream clustering [1]–[3] is challenging due to the inherent characteristics of short text steams such as short length, weak signal and high ambiguity of each short text, and the explosive growth and popularity of short textual content.

During the past decade, existing short text stream clustering has the following two kinds of methods: the one-pass-based [1], [4]–[6] and batch-based [7]–[9]. The one-pass-based method assumes that the streaming texts come one by one, we can process each text only one time. The batch-based method assumes that the streaming texts come in batch, we can process the texts in each batch multiple times. The one-pass-based method has the advantages of handling large-scale text steams efficiently. However, the main drawback is that the performance is not satisfactory because of the sparsity problem of short text. For the batch-based method, the performance can improve apparently when we allow multiple iterations of each batch. However, the real-time is relatively poor because of multiple iterations of each batch.

To overcome these inherent weaknesses and keep the advantages of both one-pass-based and batch-based methods, we propose a novel clustering method, namely Lifelong learning Augmented Short Text stream clustering method (LAST), which adds episodic memory module and sparse experience replay module of lifelong learning into existing clustering method. Specifically, LAST is augmented with an episodic memory module that randomly saves previously seen texts throughout its lifetime. In the processing of experience replay, we sample the related texts in episodic memory at a certain interval and perform cluster feature updates based on the retrieved examples. The episodic memory module in deep learning is used to address the problem of catastrophic forgetting. Here, it is used to help enhance the influence of some recent active clusters by updating their cluster feature vectors. Considering that continual texts in text streams have a higher probability of belonging to the same clustering,

The associate editor coordinating the review of this manuscript and approving it for publication was Gang Mei.

sparse experience replay is an advisable and promising strategy for short text stream clustering.

Compared with the batch-based method, the LAST-based method can effectively deal with large-scale data streams. Compare with the one-pass-based method, LAST copes with the sparsity problem of short texts by performing sparse experience replay at a certain interval. To compare the performance of the three kinds of methods, we choose three existing short text stream clustering methods to do experiments. Experimental results show that the LAST-based method obtains obvious improvement compared with the one-pass-based method on two public datasets, and is many times faster than the batch-based method. To the best of our knowledge, it is the first work to integrate lifelong learning for short text stream clustering.

The following sections are organized as follows: Section 2 describes the related work; Section 3 details the formulation and procedure of Dirichlet-based short text stream clustering; Section 4 presents our proposed method LAST; Section 5 shows the experimental results; Section 5 summarizes the paper.

II. RELATED WORK

Depending on the adopted techniques, short text stream clustering includes similarity-based and Dirichlet-based clustering. The similarity-based method often uses vector space models to represent texts and calculates the similarity between texts or clusters. The Dirichlet-based method utilizes topic modeling to learn the hidden topics. It assumes that text is generated through the process of ''Select a topic with a certain probability and select a word with a certain probability from this topic'' [2], [10]. Generally, parameters are estimated by Gibbs sampling [11] or the EM algorithm [12].

Depending on the scanning mode, short text stream clustering can be classified into two kinds of methods: batch-based and one-pass-based. The batch-based method assumes that the streaming texts come in batch, we can process the texts in each batch multiple times [13]. The one-pass-based method assumes that we can process each text only one time [4].

A. SIMILARITY-BASED SHORT TEXT STREAM CLUSTERING

CluStream [14] is one of the most classic stream clustering methods that consists of online micro-clustering and offline macro-clustering. CluStream uses a pyramid time frame to store micro-clusters at different times in the past for future analysis. Yoo *et al.* [15] proposed a streaming spectral clustering method that maintains the approximation of the normalized Laplace operator for data streams over time and effectively updates the Laplace transformed eigenvectors as streams. Shou *et al.* [4] proposed a prototype of a persistent summary for Twitter text streaming, called Sumblr which compresses tweets into tweet feature vectors (TCVs) and processes them online. Kalogeratos *et al.* [16] proposed a method for clustering text streams using burst word information. This approach makes use of the fact that most of the important documents in a topic were published during the eruption of the term ''main''.

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

B. DIRICHLET-BASED SHORT TEXT STREAM CLUSTERING

We introduce the existing Dirichlet-Based Short Text Stream Clustering methods based on the scanning model.

The batch-based methods are first studied, which iterate the texts of each batch multiple times. Yin *et al.* proposed a dynamic GSDMM [17] model that considers that all the words in a text as belonging to a single topic and effectively solves the sparsity problem of short text. The DCT [9] model assumes that topics at the previous time may have a guiding effect on topics at a later time. Therefore, the number of topics must be specified in advance. The MStream [6] method based on the Dirichlet process can automatically infer the number of topics. The NPMM [18] model is a recently introduced model that uses word embeddings to eliminate a cluster generating parameter from the model.

The one pass-based methods attracted much attention in recent years, which can cluster text streams in time. Yin *et al.* [6] proposed one-pass-based clustering method based on the Dirichlet process multinomial mixture model. Kumar *et al.* [1] integrated semantic information for topic-based clustering, which can handle ''word ambiguity'' problem effectively.

Although both the batch-based and one-pass-based methods have obvious advantages, their disadvantages are equally notable. For the batch-based methods, its real-time for dealing with large-scale text stream is usually a big question. For the one-pass-based methods, they cannot handle the sparseness problem of short texts very well. In contrast with the batch-based and one pass-based method, we aim to propose a novel short text stream clustering method, which can integrate both of their advantages. We incorporate the episodic memory module and sparse experience replay module of lifelong language learning into the clustering process. The ability to continuously learn and accumulate knowledge throughout a lifetime and reuse it effectively to adapt to a new problem quickly is a hallmark of general intelligence [19].

III. SHORT TEXT STREAM CLUSTERING

In this section, we introduce the common framework of short text stream clustering based on the Dirichlet model, denoted as Dirichlet-based clustering methods.

Formally, a text stream is continuous arrival of texts over time: $S = \{s_t\}_{t=1}^{\infty}$, with a vocabulary V, where s_t represents a text arrived at time t. The aim of the clustering task is to assign a cluster label for each text: $Z = \{z_t\}_{t=1}^{\infty}$, where z_t is the cluster label of text s_t. In general, the number of cluster labels is much less than the number of texts. For most short text stream clustering methods, they assume that each text only belongs to one topic, so $z_i \cap z_j = \emptyset$, where $i \neq j$.

Dirichlet-based clustering methods adopt the Dirichlet process to model the data. The Dirichlet process [20] is a stochastic process, most commonly used as a prior for mixture models, which is widely used in nonparametric Bayesian models. In the generative process, clusters and words are drawn from the multinomial distributions of the mixture model, and the Dirichlet process is the prior of these multinomial distributions, namely, cluster-word distribution ϕ and document-cluster distribution θ as prior distribution with parameter α and β.

The mixture weights $\theta = \{\theta\}_{k=1}^{\infty}$ can be formalized by $\theta \sim GEM(\gamma)$ [21]. In the Dirichlet process, there is no need to initialize the number of clusters in advance. The Chinese restaurant process (CRP) [22] is a popular way to explain this process. Suppose a restaurant has an infinite number of tables and initially there is no one in the restaurant. The first customer enters and chooses the first table, and the next customer chooses either the occupied table k with n_k people with a probability of $\frac{n_k}{\alpha+N-1}$ or picks an empty table with a probability of $\frac{\alpha}{\alpha+N-1}$, where N is the total number of customers in the restaurant. Therefore, at each time, new text is dynamically assigned to the cluster, similarly to the CRP.

The generative process of Dirichlet-based clustering methods is described as follows:

$$z_t \mid \theta \sim Mult(\theta) \quad t = 1, \cdots, \infty$$
$$\phi_k \mid \beta \sim Dir(\beta) \quad k = 1, \cdots, \infty$$
$$s_t | z_t, \{\phi_k\}_{k=1}^{\infty} \sim p(s_t|\phi_{z_t})$$

where Dir is a Dirichlet distribution, $Mult$ is a multinomial distribution, and z_t represents the cluster label assigned to text s_t.

The probability of text s_t generated by cluster z_t is defined as follows:

$$P(s_t|\phi_{z_t}) \propto \prod\nolimits_{w \in s_t} Mult(w \mid \phi_{z_t}) \quad (1)$$

Dirichlet-based clustering method represents a cluster with the cluster feature (CF) vector. The CF vector of a cluster z is defined as a tuple $\{n_z^w(w \in V), m_z, n_z\}$, where n_z^w is the number of frequency of word w in cluster z, m_z is the number of texts in cluster z, and n_z is the number of words in cluster z.

The CF vector presents the following two addible and deletable properties, as described next.

Addible Property. A text s_t can be efficiently added to cluster z by updating its CF vector as follows.

$$n_z^w = n_z^w + n_{s_t}^w \text{ for each word } w \text{ in } s_t$$
$$m_z = m_z + 1 \quad n_z = n_z + n_{s_t}$$

Here, $n_{s_t}^w$ is the number of occurrences of word w in text s_t, and n_{s_t} is the number of words in s_t.

Deletable Property. A text s_t can be efficiently deleted from cluster z by updating its CF vector as follows.

$$n_z^w = n_z^w - n_{s_t}^w \text{ for each word } w \text{ in } s_t$$
$$m_z = m_z - 1 \quad n_z = n_z - n_{s_t}$$

The execution process of the one-pass-based clustering methods is shown in Algorithm 1. For the first text, it will choose a new cluster. The CF vector of this newly created cluster will be initialized with the first text. Afterward, each arriving text in the stream either chooses one of the existing clusters or generates a new cluster according to the probability computed by the Dirichlet-based clustering method. When a new cluster is chosen, the method will create a new cluster to store the corresponding cluster feature vector. Otherwise, the method will add the corresponding text into the chosen existing cluster with the addible property.

Algorithm 1 The One-Pass-Based Clustering Method

Input: $S = \{s_t\}_{t=1}^{\infty}$
Output: Cluster label $\{z_t\}_{t=1}^{\infty}$

```
1: $K \leftarrow \emptyset$
2: for $t = 1$ to $\infty$ do
3:     for Each existing clusters do
4:         Compute the probability of $s_t$
5:     end for
6:     Compute the probability of $s_t$ choosing a new cluster
7:     Sample cluster index $z_t$ for text $s_t$
8:     Updating the CF vector of cluster $z_t$ with the addible
   property
9: end for
```

The execution process of the batch-based clustering methods is shown in Algorithm 2. The batch-based method assumes that the streaming texts come in batch, which can process the texts in each batch multiple times. When a batch of texts comes, the batch-based method first obtains an initial cluster label using the above one-pass-based method. From the second iteration, the batch-based method first deletes it from its current cluster with the deletable property. Then, it reassigns the text to a cluster according to the probability of the text.

IV. SHORT TEXT STREAM CLUSTERING WITH EPISODIC MEMORY

Considering the inherent characteristics of short text steams, we introduce a novel short text clustering method, namely Lifelong learning Augmented Short Text Stream Clustering (LAST), to alleviate the sparseness problem of the short text streams. Our LAST-based method is illustrated in Figure 1. Compared with the one-pass-based method, we add two novel modules (episodic memory and sparse experience replay) into short text stream clustering. Our method LAST is augmented with an episodic memory module that saves previously seen texts throughout its lifetime at a certain interval. Experience replay that randomly chooses the recent active texts from memory help to enhance the effect of the current clustering.

A. EPISODIC MEMORY

The episodic memory module is a block that stores some previously seen texts. If we assume that the model has unlimited

Algorithm 2 The Batch-Based Clustering Method

Input: $S = \{s_i\}_{i=1}^{N}$ of batch t
Output: Cluster label $\{z_i\}_{i=1}^{N}$ of batch t

```
1:  for iter = 1 do
2:      Obtain an initial clustering results for $\{s_i\}_{i=1}^{N}$ using
        the one-pass clustering process
3:  end for
4:  for iter = 2 to $I$ do
5:      for $i = 1$ to $N$ do
6:          Record the current cluster $z_i$ of $s_i$
7:          Updating the CF vector of cluster $z_i$ with the
            deletable property
8:          for Each existing clusters do
9:              Compute the probability of $s_i$
10:         end for
11:         Compute the probability of $s_i$ choosing a new
            cluster
12:         Sample cluster index $z_i$ for text $s_i$
13:         Updating the CF vector of cluster $z_i$ with the
            addible property
14:     end for
15: end for
```

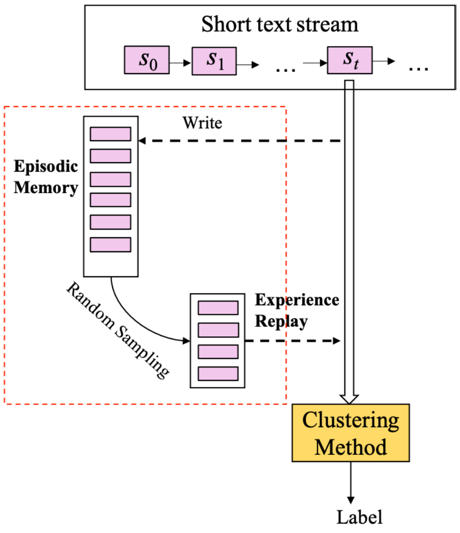


FIGURE 1. Diagram of LAST model.

capacity, we can store all streaming texts into the memory. However, this assumption is unrealistic in practice, because it violates the essence of short text stream clustering. We adopt a simple writing strategy that writes a new text into the memory at a certain interval. It is a common phenomenon that continual texts in text streams have a higher probability of belonging to the same clustering. The memory only saves the latest M texts and automatically deletes outdated texts.

B. SPARSE EXPERIENCE REPLAY

Because each arriving text in the stream is only used once, the one-pass-based short text clustering method suffers from the lack of statistical information to capture semantics. Therefore, at a certain interval during the process of clustering, we sample the texts from the episodic memory for experience replay. Since the cluster feature vectors of some clusters are diminished over time, experience replay will help enhance the influence of the recent clusters. If we allow the model to perform experience replay at every timestep, it will consume much time and violates our purpose. Therefore, we only sample part of the texts to perform sparse experience replay. We adopt random sampling to perform sparse experience replay. In practice, we randomly retrieve 60 texts to update their CF features every 30 new texts. We only perform one-pass scanning for the 60 retrieved texts. Besides, each text of them only chooses one of the existing clusters instead of a new cluster, because all of them are outdated texts.

C. LAST METHOD

In the subsection, we will describe the details of our proposed LAST. In the process of sparse experience replay, we set up a certain interval throughout the clustering period for sampling stored examples in the memory (lines 3-6). LAST infers the hidden topics for the current text using any one of the short text stream clustering methods (lines 7-12). Every five texts are processed, the 5th text is written into the episodic memory (lines 13-15).

Algorithm 3 The LAST-Based Clustering Method

Input: $S = \{s_t\}_{t=1}^{\infty}$, replay interval R and store interval E.
Output: Cluster label z_{s_t} for each text.

```
1:  $K \leftarrow \emptyset$
2:  for $t = 1$ to $\infty$ do
3:      if $t\%R = 0$ then
4:          Sample $S$ examples from memory $M$.
5:          Perform experience replay using clustering
            method.
6:      end if
7:      for Each existing clusters do
8:          Compute the probability of $s_t$.
9:      end for
10:     Compute the probability of $s_t$ choosing a new cluster.
11:     Sample cluster index $z_{s_t}$ for text $s_t$.
12:     Updating the CF vector of cluster $z_{s_t}$ with the addible
        property.
13:     if $t\%E = 0$ then
14:         Write $\{s_t\}$ into Memory M and delete old texts in
            M.
15:     end if
16: end for
```

D. COMPLEXITY ANALYSIS

We compare the counts of scanning of the three kinds of short text stream clustering methods: batch-based, one-pass-based, and LAST-based. The difference between them is the time of scanning the texts. Suppose there are N texts in the stream texts. The counts of scanning in the one-pass-based method are N. For a batch-based method, it needs to iterate many times for each batch of texts. The counts of scanning in the batch-based method are $I \times N$, where I is the number of iterations. Here, I is usually a large number, at least larger than 10. The counts of scanning in the proposed LAST-based method are $N + \lfloor N/R \rfloor \times E$), where R is the replay interval and E is the number of retrieved texts from memory. In this paper, R and E are set to 30 and 60, respectively. Let $N = 1000$ and $I = 10$. The counts of scanning in one-pass-based, batch-based, and LAST-based methods are 1000, 10000, and 2980, respectively.

V. EXPERIMENTS

In this section, we evaluate the performance of our proposed method by comparison with batch-based and one-pass-based methods on three state-of-the-art short text stream clustering methods, and design experiments to answer the following questions:

Q1. The effectiveness of the LAST-based method: Do the proposed LAST-based method outperform the one-pass-based methods?

Q2. The running time of the LAST-based method: Do the running time of the proposed LAST-based method outperforms the batch-based method?

Q3. The factors affecting the LAST-based method: Experiments on different parameters verify the impact on the LAST-based method.

A. EXPERIMENTAL SETUP

1) DATASETS

We choose two short text stream datasets for experiments. The two datasets have been preprocessed by word segmentation, stop word removal, lowercase conversion, etc., and the average length matches the size of the short text.

- **News**: This dataset is the GoogleNews dataset used in GSDMM [23]. News contains 11109 news titles belonging to 152 topics, with an average length of 6.23.
- **Tweet**: This dataset is the public dataset Tweets [6], which includes 30,322 tweets that are closely related to the 269 query items in TREC 2011-2015 microblog tracking. The average length of tweets in the dataset is 7.97.

2) EVALUATION METRICS

We employ five widely used metrics to evaluate the clustering performance: normalized mutual information (NMI), purity, accuracy, homogeneity, and completeness [24]–[26].

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

$$NMI = \frac{\sum_c \sum_k n_{c,k} \log(\frac{N \cdot n_{c,k}}{n_c \cdot n_k})}{\sqrt{\sum_c n_c \log(\frac{n_c}{N}) \sum_k n_k \log(\frac{n_k}{N})}} \tag{2}$$

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

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

$$Purity = \frac{1}{N} \sum_k \max_c |n_c \cap n_k| \tag{3}$$

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

$$Accuracy = \frac{1}{N} \sum\nolimits_i^N \delta(c_i, map(k_i)) \tag{4}$$

where k_i and c_i represent the clustering result and the real label corresponding to data x_i respectively, $map(k_i)$ denotes the optimal class label distribution, and the Hungarian algorithm [27] is used to achieve the optimal mapping. In addition, $\delta(a, b)$ is the indicator function. If $a = b$, the value is 1, otherwise it is 0.

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

$$Homogeneity = 1 - \frac{H(C|K)}{H(C)} \tag{5}$$

where $H(C|K)$ is the conditional entropy of the class assigned to a given cluster, and $H(C)$ is the class entropy [28].

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

$$Completeness = 1 - \frac{H(K|C)}{H(K)} \tag{6}$$

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

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

3) METHODS FOR COMPARISON

We choose the following three baselines to verify three kinds of methods: batch-based, one-pass-based, and LAST-based:

MStream. MStream [6] is based on the Dirichlet process multinomial mixture model. It includes the one-pass-based method and batch-based method. The number of iterations in a batch-based method is set to 10. When the number of iterations is set to 1, it is changed to the one-pass-based method.

MStreamF. MStreamF [6] is a variation of MStream, which clusters text streams by time point with forgetting rules. Only texts within a limited time range are stored in memory. The parameters of MStream and MStreamF in the original paper are set $\alpha = 0.03$ and $\beta = 0.03$. For the batch-based method, it sets the maximum storage batch to 2 batches, and the number of iterations to 10.

TABLE 1. Experimental results of all models on two datasets.

Datasets		News					Tweet				
Method		NMI	H	C	P	A	NMI	H	C	P	A
MStream	Batch	**0.855**	0.843	**0.868**	0.751	**0.716**	**0.842**	0.828	**0.841**	0.689	0.608
	One-pass	0.807	0.820	0.795	0.716	0.656	0.809	0.826	0.792	0.733	0.624
	LAST	0.84	**0.848**	0.831	**0.759**	0.697	0.84	**0.856**	0.825	**0.773**	**0.663**
MStreamF	Batch	0.771	0.742	0.802	0.644	0.617	**0.822**	0.786	**0.859**	0.637	0.597
	One-pass	0.708	0.706	0.710	0.608	0.550	0.803	0.792	0.813	**0.685**	0.607
	LAST	**0.798**	**0.787**	**0.81**	**0.707**	**0.67**	0.819	**0.802**	0.837	**0.685**	**0.624**
OSDM	Batch	**0.779**	0.794	**0.766**	0.654	**0.508**	**0.845**	0.959	0.737	0.928	0.507
	One-pass	0.745	0.749	0.742	0.601	0.471	0.837	0.935	**0.749**	0.889	**0.524**
	LAST	0.765	**0.802**	0.729	**0.671**	0.490	0.840	**0.961**	0.734	**0.930**	0.517

TABLE 2. The running time of all models on two datasets.

Datasets		News	Tweet
MStream	Batch	682.13	3048.43
	One-pass	46.61	247.05
	LAST	97.13	585.62
MStreamF	Batch	429.47	1558.67
	One-pass	53.60	195.98
	LAST	92.12	260.17
OSDM	One-pass	52.07	195.13
	LAST	142.27	552.53

OSDM. OSDM is a one-pass-based method, which integrates the word-occurrence semantic information into the MStream method. We follows the original parameters ($\alpha = 2e^{-3}$, $\beta = 4e^{-5}$ and $\gamma = 6e^{-6}$) used by the authors. To compare the three methods, we add one batch-based method for OSDM. The number of iterations is also set to 10.

LAST. For the LAST-based method, replay interval R and store interval E are set to 30 and 60, and the size of episodic memory is 500.

B. EVALUATION OF THREE KINDS OF METHODS

The results are shown on Table 1. As can be seen, LAST-based methods outperform one-pass-based methods in most cases, which verify that the one-pass-based methods can be improved by incorporating a lifelong learning mechanism, because sparse experience replay helps alleviate the sparseness problem. Compared with the batch-based methods, LAST-based methods still achieve good results in many metrics, since the texts in each batch are fixed in the batch-based method, and the texts sampled by sparse experience replay in the LAST-based method are dynamic which will bring diversity to the learning process. When adopting the MStreamF method, we can see that the LAST-based method outperforms the batch-based method on all metrics on the News corpus.

In conclusion, The LAST-based methods can address the sparsity problem and obtain good clustering results by accounting for the lifelong learning mechanism.

We also compare the running time of all models on two datasets. The experimental results are shown on Table 2. We see that the one-pass methods achieve the lowest running time and the batch-based methods are the slowest. The running times of the LAST-based methods are only double or triple longer than the one-pass-based methods. The LAST-based methods are more effective than the batch-based methods. The experimental results are per the complexity analysis.

C. INFLUENCE OF DATA SCALE

In this subsection, we will verify the influence of data scale on the cluster performance. We will choose the newest OSDM as a comparison model, and choose the different percentages of the News and Tweet datasets as a subset to do experiments. We set the percentages varying from 10% to 100%.

The experimental results between the LAST-based OSDM (LAST) and the original one-pass-based OSDM (OSDM) are shown on Tabel 3 and Tabel 4.

We can see that LAST significantly outperforms OSDM in most cases. As the dataset becomes smaller, the sparseness problem will have an increasing impact on the performance of the clustering methods. Compare with OSDM, LAST copes with the sparsity problem of short texts by performing sparse experience replay at a certain interval. It suggests that the lifelong learning of LAST is a very effective means for alleviating sparseness.

D. ABLATION STUDY OF LAST

To further analyze the factors affecting the LAST-based approaches, we do more experiments in this section. We analyze the three parameters the size of memory (M), replay interval (R), and the number of retrieving texts from memory (E). For each experiment, we vary one parameter and fix the other two parameters. To a better comparison, we show the results of the one-pass-based method and the batch-based method as a reference. Here, we only choose one dataset (News) to do the analysis.

(1) Influence of the size of memory (M)

We conducted experiments to set the number of the size of memory M varying from 100 to 100 under fixing the other parameters ($R = 30$ and $E = 60$). Figure 2 shows the NMI values of the three LAST-based clustering methods on the News dataset.

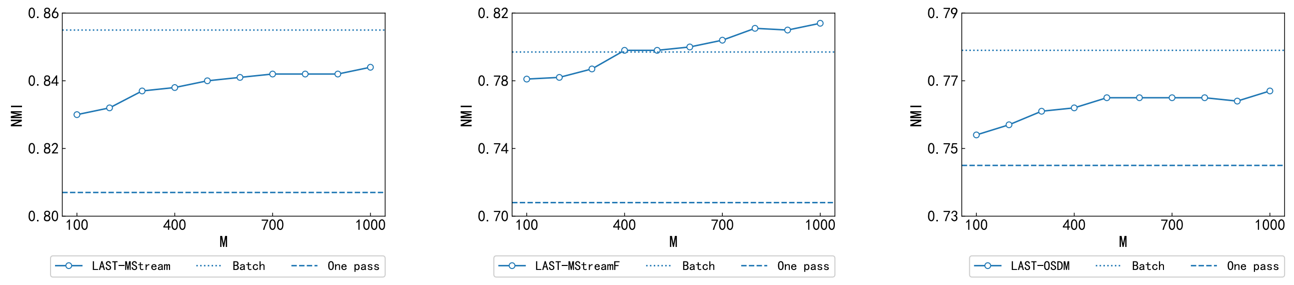
We can see that the NMI value increases gradually when M is changed from 100 to 1000. As M increases, it means that more recent texts are stored into the memory module, which makes the topic distribution of the sampled texts

TABLE 3. Clustering results on news subsets using OSDM.

Metric		10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
NMI	OSDM	0.722	0.735	0.745	0.750	0.749	0.753	0.756	0.757	0.757	0.745
	LAST	**0.745**	**0.758**	**0.763**	**0.765**	**0.763**	**0.762**	**0.763**	**0.764**	**0.765**	**0.765**
H	OSDM	0.643	0.676	0.695	0.704	0.707	0.715	0.723	0.730	0.734	0.749
	LAST	0.706	0.750	0.771	0.780	0.784	0.787	0.793	0.797	0.801	0.802
C	OSDM	**0.810**	**0.798**	**0.799**	**0.799**	**0.793**	**0.793**	**0.790**	**0.786**	**0.780**	**0.742**
	LAST	0.787	0.767	0.755	0.750	0.743	0.737	0.735	0.733	0.731	0.729
Purity	OSDM	0.428	0.489	0.516	0.530	0.542	0.555	0.566	0.576	0.584	0.601
	LAST	**0.491**	**0.565**	**0.606**	**0.623**	**0.635**	**0.643**	**0.653**	**0.660**	**0.667**	**0.671**
CA	OSDM	0.409	0.456	0.475	**0.483**	**0.487**	**0.495**	**0.498**	**0.499**	**0.500**	0.471
	LAST	**0.427**	**0.459**	**0.476**	0.479	0.482	0.481	0.484	0.485	0.489	**0.490**

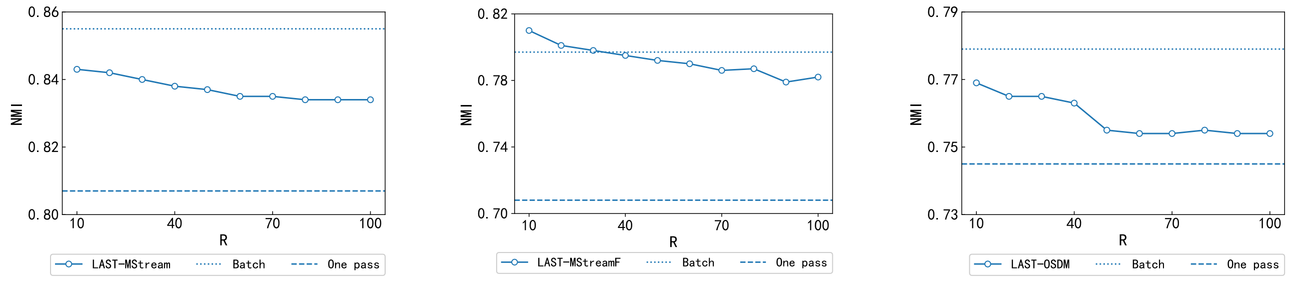
TABLE 4. Clustering results on tweet subsets using OSDM.

Metric		10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
NMI	OSDM	0.777	0.773	0.798	0.800	0.808	0.818	0.828	0.839	**0.843**	0.837
	LAST	**0.789**	**0.785**	**0.801**	**0.826**	**0.831**	**0.836**	**0.840**	**0.844**	**0.843**	**0.840**
H	OSDM	0.772	0.855	0.889	0.899	0.902	0.910	0.922	0.931	0.937	0.935
	LAST	**0.875**	**0.904**	**0.923**	**0.939**	**0.942**	**0.946**	**0.951**	**0.956**	**0.959**	**0.961**
C	OSDM	**0.783**	**0.698**	**0.715**	0.713	0.724	0.736	0.743	**0.756**	**0.758**	**0.749**
	LAST	0.712	0.682	0.694	**0.727**	**0.734**	**0.740**	**0.741**	0.745	0.741	0.734
Purity	OSDM	0.769	0.857	0.856	0.850	0.848	0.855	0.870	0.883	0.891	0.889
	LAST	**0.805**	**0.853**	**0.879**	**0.900**	**0.901**	**0.902**	**0.912**	**0.920**	**0.924**	**0.930**
CA	OSDM	**0.567**	**0.543**	**0.511**	0.485	0.494	0.510	0.514	0.536	**0.534**	0.524
	LAST	0.522	0.489	0.501	**0.531**	**0.536**	**0.531**	**0.531**	**0.539**	0.523	**0.517**



(a). MStream (b). MStreamF (c). OSDM

FIGURE 2. Influence of the size of memory for the LAST-based methods.



(a). MStream (b). MStreamF (c). OSDM

FIGURE 3. Influence of replay interval for the LAST-based methods.

more reasonable, and makes the updating of the cluster feature tuple more reasonable when executing the experience replays.

(2) Influence of replay interval (*R*)

By varying the replay interval *R*, we study the effect of *R* in LAST. We set *R* varying from 10 to 100 under fixing the other parameters ($M = 500$ and $E = 60$). Figure 3 shows the NMI values of the three LAST-based clustering methods on News dataset.

With the increase of *R*, the number of sparse experience replay gradually decreases, which weakens the influence of LAST during the process of clustering. When *R* increases to a certain extent (greater than the number of texts in the dataset), the LAST-based method becomes the one-pass-based method. We see that the NMI values decrease gradually with the increase of *R*. Since the cluster feature vectors of some clusters are diminished over time, experience replay will help enhance the influence of the recent clusters. But,

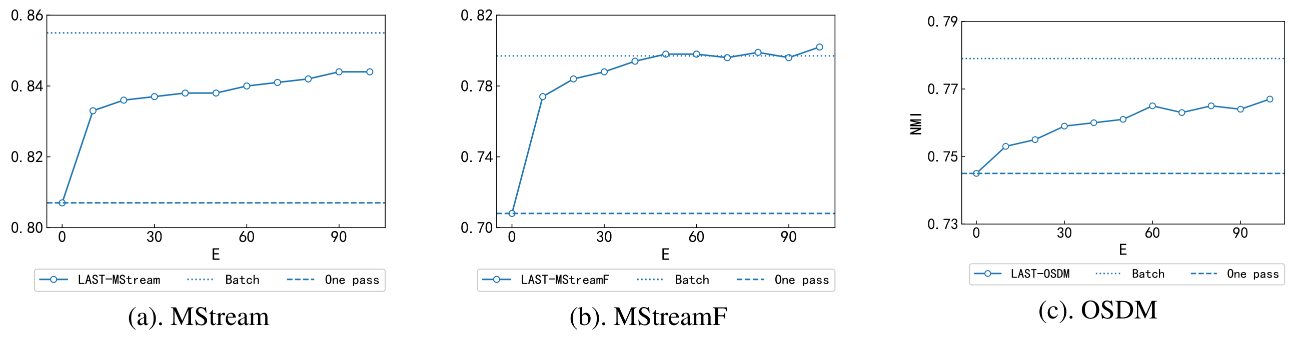


FIGURE 4. Influence of the number of retrieve texts for the LAST-based methods.

we cannot perform experience replay at every timestep, it will spend much time. Therefore, we only perform sparse experience replay at a certain interval.

(3) Influence of the number of retrieve texts from memory (E)

By varying the number of retrieving texts from memory E, we study the effect of E for LAST. We set E varying from 0 to 100 under fixing the other parameters ($M = 500$ and $R = 30$).

Figure 4 shows the results on the News dataset. When E equals 0, the LAST-based method is equivalent to the one-pass-based method. With the increase of E, we sample more texts each time to update the cluster feature vectors in experience replay. In the early stage of clustering, the one-pass-based method cannot deal with the sparseness very well. More texts used in experience replay can help to alleviate the sparsity of a short text stream.

VI. CONCLUSION

In contrast to the existing batch-based method and one-pass-based method, we proposed a novel Lifelong learning Augmented Short Text stream clustering method (LAST), which incorporates the lifelong learning mechanism into the short text stream clustering method. LAST also processes each text one time, but at a certain interval, it randomly samples some recent texts of episodic memory to update cluster features by performing sparse experience replay. LAST can alleviate the sparseness problem. Since the cluster feature vectors of some clusters are diminished over time, experience replay will help enhance the influence of the recent clusters. Experimental results demonstrate that LAST achieves better performance with lower complexity on two datasets using five metrics compared with the batch-based and one-pass-based methods. In the future, we will explore more retrieval mechanisms to perform sparse experience replay, e.g., K-nearest neighbors.

REFERENCES

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

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

[3] J. Qiang, Z. Qian, Y. Li, Y. Yuan, and X. Wu, "Short text topic modeling techniques, applications, and performance: A survey," *IEEE Trans. Knowl. Data Eng.*, early access, May 4, 2020, doi: 10.1109/TKDE.2020.2992485.

[4] L. Shou, Z. Wang, K. Chen, and G. Chen, "Sumblr: Continuous summarization of evolving tweet streams," in *Proc. 36th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2013, pp. 533–542.

[5] F. Cao, M. Estert, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2006, pp. 328–339.

[6] J. Yin, D. Chao, Z. Liu, W. Zhang, X. Yu, and J. Wang, "Model-based clustering of short text streams," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2634–2642.

[7] A. Ahmed and E. Xing, "Dynamic non-parametric mixture models and the recurrent Chinese restaurant process: With applications to evolutionary clustering," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2008, pp. 219–230.

[8] H. Amoualian, M. Clausel, E. Gaussier, and M.-R. Amini, "Streaming-LDA: A copula-based approach to modeling topic dependencies in document streams," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 695–704.

[9] S. Liang, E. Yilmaz, and E. Kanoulas, "Dynamic clustering of streaming short documents," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 995–1004.

[10] J. Qiang, Y. Li, Y. Yuan, and W. Liu, "Snapshot ensembles of non-negative matrix factorization for stability of topic modeling," *Appl. Intell.*, vol. 48, no. 11, pp. 3963–3975, 2018.

[11] A. Terenin, D. Simpson, and D. Draper, "Asynchronous Gibbs sampling," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 144–154.

[12] D. Chu, M. Reyers, J. Thomson, and L. Y. Wu, "Route identification in the national football league: An application of model-based curve clustering using the EM algorithm," *J. Quant. Anal. Sports*, vol. 16, no. 2, pp. 121–132, Jun. 2020.

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

[14] C. C. Aggarwal, S. Y. Philip, J. Han, and J. Wang, "A framework for clustering evolving data streams," in *Proc. VLDB Conf.*, 2003, pp. 81–92.

[15] S. Yoo, H. Huang, and S. P. Kasiviswanathan, "Streaming spectral clustering," in *Proc. IEEE 32nd Int. Conf. Data Eng. (ICDE)*, May 2016, pp. 637–648.

[16] A. Kalogeratos, P. Zagorisios, and A. Likas, "Improving text stream clustering using term burstiness and co-burstiness," in *Proc. 9th Hellenic Conf. Artif. Intell.*, May 2016, pp. 1–9.

[17] J. Yin and J. Wang, "A text clustering algorithm using an online clustering scheme for initialization," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1995–2004.

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

[19] C. de Masson d'Autume, S. Ruder, L. Kong, and D. Yogatama, "Episodic memory in lifelong language learning," 2019, *arXiv:1906.01076*. [Online]. Available: http://arxiv.org/abs/1906.01076

[20] Y. W. Teh, "Dirichlet process," Tech. Rep., 2010.

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

[22] D. M. Blei, T. L. Griffiths, and M. I. Jordan, "The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies," *J. ACM*, vol. 57, no. 2, pp. 1–30, Jan. 2010.

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

[24] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "Hybrid clustering analysis using improved krill herd algorithm," *Appl. Intell.*, vol. 48, no. 11, pp. 4047–4071, 2018.

[25] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "A new feature selection method to improve the document clustering using particle swarm optimization algorithm," *J. Comput. Sci.*, vol. 25, pp. 456–466, Mar. 2018.

[26] L. M. Q. Abualigah, *Feature Selection and Enhanced Krill Herd Algorithm for Text Document Clustering*. Springer, 2019.

[27] G. A. Mills-Tettey, A. Stentz, and M. B. Dias, "The dynamic Hungarian algorithm for the assignment problem with changing costs," Robot. Inst., Pittsburgh, PA, USA, Tech. Rep. CMU-RI-TR-07-27, 2007.

[28] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *Proc. Joint Conf. Empirical Methods Natural Lang. Process. Comput. Natural Lang. Learn. (EMNLP-CoNLL)*, 2007, pp. 410–420.

JIPENG QIANG received the Ph.D. degree in computer science and technology from the Hefei University of Technology, in 2016. He was a Ph.D. Visiting Student with the Artificial Intelligence Laboratory, the University of Massachusetts Boston, from 2014 to 2016. He is currently an Associate Professor with the School of Information Engineering, Yangzhou University. He has received two grants from the National Natural Science Foundation of China, one grant from the Natural Science Foundation of Jiangsu Province of China, one grant from the Natural Science Foundation of the Higher Education Institutions of Jiangsu Province of China. He has published more than 40 articles, including *AAAI*, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, *TKDD*, and *TASL*. His research interests include topic modeling and natural language processing.

WANYIN XU received the B.S. degree in computer science from Yangzhou University, China, where he is currently pursuing the M.S. degree in computer science. His research interests include topic modeling and data mining.

YUN LI received the M.S. degree in computer science and technology from the Hefei University of Technology, China, in 1991, and the Ph.D. degree in control theory and control engineering from Shanghai University, China, in 2005. He is currently a Professor with the School of Information Engineering, Yangzhou University, China. He has published more than 100 scientific articles. His research interests include data mining and cloud computing.

YUNHAO YUAN received the M.Eng. degree in computer science and technology from Yangzhou University, China, in 2009, and the Ph.D. degree in pattern recognition and intelligence systems from the Nanjing University of Science and Technology, China, in 2013. He is currently an Associate Professor with the School of Information Engineering, Yangzhou University. His research interests include pattern recognition, data mining, and image processing.

YI ZHU received the B.S. degree from Anhui University, the M.S. degree from the University of Science and Technology of China, and the Ph.D. degree from the Hefei University of Technology. He is currently an Assistant Professor with the School of Information Engineering, Yangzhou University, China. His research interests include data mining and knowledge engineering, and recommendation systems.