2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) | 979-8-3503-8622-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/BIBM62325.2024.10822727

Efficient Topology-Driven Clustering for Imbalanced Streaming Biomedical Data Analysis

Xiaopeng Luo[1,2], Yiqun Zhang[2]✉, Yuzhu Ji[2], Peng Liu[2], Taoting Xiao[3]
[1]*Guangzhou Huali College*, [2]*Guangdong University of Technology*, [3]*Guangzhou College of Commerce*

***Abstract*—Clustering drifting data is common in the field of biomedical data analysis. Data chunks collected at different periods often exhibit clusters with significantly different sizes, and drifting distributions of clusters also appear frequently. We call such composite phenomenon imbalance-drifting, which can severely impact the accuracy and efficiency of cluster analysis. Therefore, we propose a topology-representation-based clustering paradigm, which first learns an informative global data representation in a self-organizing manner to obtain a map with nested representative data points. Then fast and accurate clustering is facilitated by quickly retrieving similar data points according to the topology. As the constructed Self-Organizing Map (SOM) is exploited for informative representation, micro partition, and quick merging, to achieve advanced clustering under imbalance-drifting, the proposed approach is thus called Tri-Squeezing SOM for Clustering (TSSC). It turns out that TSSC significantly reduces the time complexity for clustering an n-scale imbalance-streaming data without sacrificing accuracy. Moreover, TSSC can automatically determine the number of clusters k, and features interpretability and hyper-parameter robustness. Extensive results on both biomedical datasets and synthetic datasets verify the superiority of TSSC.**

***Index Terms*—Streaming biomedical big data, concept drift, imbalanced data, cluster analysis, efficient algorithms**

I. Introduction

Biomedical drifting data is prevalent in the machine learning domain related to biomedicine [1], particularly in streaming data scenarios, the statistical properties of the data such as distributions and label proportions may change over time [2]. This phenomenon is referred to as "drifting". Besides, the changing distribution of samples often results in imbalanced clusters, referred to as Imbalance-Drifting Data (IDD) [3]. Clustering, a key unsupervised machine learning technique, is essential for analyzing drifting data due to the lack of labels [4]. However, the existing imbalanced data clustering approaches [5] are extremely time-consuming, while fast clustering methods for streaming data [6] are incompetent in exploring imbalanced clusters. Such an intractable trade-off poses great challenges to the analysis of IDD biomedical data.

Due to the frequent incidence of cluster size variations and concept drifts, clusters may experience imbalance-drifting.

This work was supported in part by the NSFC under Grants: 62476063, 62374047, 62302104, 62174038, and 62102097, the Natural Science Foundation of Guangdong Province under grants 2023A1515012855 and 2023A1515012884, the Science and Technology Program of Guangzhou under grant SL2023A04J01625, and the General University Youth Innovation Talent Program of Guangdong under grant 2024KQNCX094. Yiqun Zhang is the corresponding author (E-mail: yqzhang@gdut.edu.cn).

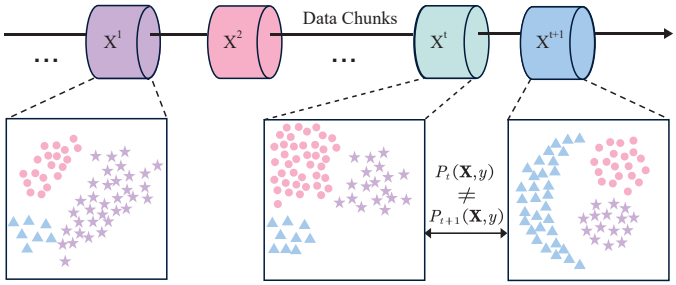

Fig. 1. A toy example of Imbalance-Drifting Data (IDD).

Fig. 1 demonstrates a representative IDD case, where imbalanced ratio $IR = \mathcal{M}/\mathfrak{m}$ with $\mathcal{M}$ and $\mathfrak{m}$ referring to the sizes of majority and minority clusters, respectively. Joint probability $P(\mathbf{X}, y)$ of data chunk $\mathbf{X}$ and cluster label y can jointly reflect the occurrence of imbalance-drifting. That is, when IR_t is relatively large and $P_{t+1}(\mathbf{X}, y) \neq P_t(\mathbf{X}, y)$, an imbalance-drifting has occurred between time-stamps t and $t+1$. Since existing efficiency-focused algorithms [7]–[9] implicitly assume the balance of clusters and data summarization [10], [11] is usually adopted to accelerate the clustering process, the well-known 'uniform effect' that partitions data samples into similar-sized clusters will be more likely to occur. In contrast, imbalanced data clustering algorithms [5], [12]–[15] excessively partition data samples into an excessive number of micro-clusters to avoid overlooking small clusters, but the resulting high computational overhead is intolerable for adapting frequently occurring concept drifts in streaming data analysis. We further analyze the clustering approaches focusing on efficiency and data imbalance below.

Efficiency-focused clustering approaches [6], [7], [16] commonly focus on processing streaming data with lower space and time consumption. StreamKM++ [8] maintains a summary of stream data, and implements clustering in the summarized data to save computation cost. However, StreamKM++ tends to generate balanced clusters, which makes them incompetent to the imbalanced clusters. The variant of the DPC method, SNN-DPC [9], accelerates the laborious cluster centers finding of DPC [17] to make it feasible to be applied to streaming data clustering. However, they suffer from the non-trivial pre-selection of hyper-parameters. To solve this problem, AMD-DPC [18] has been proposed, adopting a graph-based localized density updating strategy for neighbor selection.

This algorithm saves computation cost while still remaining accurate. Since all the above-mentioned efficiency-focused clustering approaches somewhat overlook the common cluster imbalance-drifting issue, their clustering performance will degenerate when processing under the IDD scenario.

Existing imbalanced data clustering approaches are usually built on conventional partitional clustering algorithms [12], [13], by applying them with a larger pre-set number of clusters k to obtain many micro-clusters, which are expected to be smaller than minority clusters. Then the excessive number of micro-clusters are gradually merged to form imbalanced clusters. SMCL [5] is a representative imbalanced data clustering approach, it employs a gradual addition strategy to assign seed points and adopts a competitive-penalized mechanism to more appropriately form micro-clusters. LDPI [14] utilizes a sub-cluster generation scheme to identify noise and achieves more stable performance. MCNS [15] improves convergence speed by introducing a reconstruction rate to select key clusters. However, all these approaches involve time complexity of $O(n^2)$ for n-scale dataset, which prevents them from being utilized in IDD clustering.

This paper, therefore, proposes a topology-representation-based drifting data clustering paradigm, called Tri-Squeezing SOM for Clustering (TSSC), which thoroughly exploits Self-Organizing Map (SOM) in all three phases: i) representation learning, ii) micro partition, and iii) quick merging. TSSC first learns a self-organized topology with nested representative data points to represent the whole data distribution. It then fine-tunes the topology to obtain micro-clusters that ensure the detection of imbalanced minority clusters. Finally, it merges the micro-clusters by quickly retrieving the most similar pair through the learned topology. It turns out that TSSC has lower time complexity $O(n \log n)$, very competitive clustering accuracy, and a highly interpretable process. Moreover, TSSC can automatically determine a proper cluster number k, and is robust to the settings of the other hyper-parameters. All the above merits have been comprehensively validated by the experiments. Three main contributions are summarized below:

- A new clustering paradigm called TSSC is proposed for IDD. It exploits the representation and retrieval information provided by the finely trained SOM, and achieves fast and accurate performance for complex streaming data.
- A rapid density retrieval strategy and its corresponding measure are formulated. They serve to assess the density of clusters for quick merging guided by the trained self-organized representation topology (i.e., SOM).
- An Imbalance-Drifting Data Generator (IDD-Gen) is designed to mimic the realistic generation of IDD. It can also generate extreme imbalance-drifting cases for beyond-reality harsh evaluation, and can be utilized as a general experimental tool in this field.

II. Proposed Method

Given a dataset $S = \left\{\mathbf{X}^i\right\}_{i=1}^{N}$ comprising a series of data chunks as Fig. 1. Each data chunk consists of an n-dimensional attribute vector, denoted as $\mathbf{X}^i = \left[\mathbf{x}_j^i\right]_{j=1}^{n}$. Here, n represents the number of data points in each chunk, and $\mathbf{X}^i$ has d attributes in its attribute vector $F^i = \{\mathbf{f}_1^i, \mathbf{f}_2^i, \cdots, \mathbf{f}_d^i\}$.

When obtaining a chunk $\mathbf{X}^i$ that is a part of IDD, it will serve as the metric space for the clustering. For any choice of $Z^i = \{\mathbf{z}_1^i, \mathbf{z}_2^i, \cdots, \mathbf{z}_k^i\} \subset \mathbf{X}^i$ consisting of k cluster centers, a partition of $\mathbf{X}^i$ into k clusters can be written as $\Phi(k) = \left\{C_1^i, C_2^i, \ldots, C_k^i\right\}$. Among them, C_γ^i represents the γ-th cluster, which contains all points in $\mathbf{X}^i$ that are closer to $\mathbf{z}_\gamma^i$ than to other cluster centers. Our goal is to determine the optimal number and distribution of clusters for minimizing the total distance between data points and cluster centers. In the following sections, the research goals will be approached from three different perspectives.

A. TR: Topology Representation

To expedite the rapid construction of micro-clusters, we adopt the Randomized Self-Organizing Map (RSOM) [19]. We assume that the mapping grid comprises $Q \times Q$ neurons $\mathfrak{N} = \left\{\mathfrak{n}_1, \mathfrak{n}_{2,}.....,\mathfrak{n}_{Q^2}\right\}$, while the input imbalance-drifting data chunk $\mathbf{X}^i \in \mathbb{R}^{n \times d}$ functions as the dataset. These neurons encode the relative distances between data points, where adjacent samples in the input space are mapped to adjacent output neurons, typically arranged in a regular grid (rectangular or hexagonal). In this paper, we utilize a method involving fast Poisson disk sampling and the Lloyd relaxation scheme to randomly place neurons, giving them a specific spectral distribution (blue noise). Furthermore, a set of neighbors can be defined as $\Omega\left(\mathfrak{n}_q, t\right)$, representing a group of neurons close to $\mathfrak{n}_q$ on the grid at step t.

During training, for each $\mathbf{x}_j^i$, get the nearest neuron $\bar{\mathfrak{n}}$:

$$\bar{\mathfrak{n}} = \min_{\mathfrak{n}_q \in \mathfrak{N}} distance(\mathfrak{n}_q, \mathbf{x}_j^i), \tag{1}$$

then the neurons in $\Omega\left(\bar{\mathfrak{n}}, t\right)$ will be updated according to $\Delta\mathfrak{n}_q$:

$$\Delta\mathfrak{n}_q = \varepsilon\left(t\right) h_\sigma\left(t, \mathfrak{n}_q, \bar{\mathfrak{n}}\right)\left(\mathbf{x}_j^i - \mathfrak{n}_q\right), \tag{2}$$

$\varepsilon\left(t\right)$ is the learning rate, which decreases over time, $h_\sigma\left(t, \mathfrak{n}_q, \bar{\mathfrak{n}}\right)$ is the Gaussian kernel function used to evaluate the similarity coefficient between $\mathfrak{n}_q$ and $\bar{\mathfrak{n}}$. When the RSOM iterates to convergence, the topological map $G = \{\mathfrak{N}, \mathbf{A}\}$ is obtained, where $\mathbf{A} = [a_{ij}] \in R_{Q^2 \times Q^2}$ is the adjacency matrix of neurons defined as follows:

$$a_{ij} = \begin{cases} 1, \text{if } \mathfrak{n}_i \text{ and } \mathfrak{n}_j \text{ are connected} \\ 0, \text{otherwise.} \end{cases} \tag{3}$$

Because the topology map presents a network structure, the distribution of neurons will be affected by their neighboring neurons, leading them to be unable at the centers of the representation data points. So $\mathfrak{N}$ will be used as the initial cluster centroids for k-means fine-tuning, which will position each neuron at the center of the micro-cluster, referred to as the micro-cluster centers $Z = \left\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{Q^2}\right\}$.

B. TDP: Topology-Distilled Partition

After obtaining Z, some micro-cluster centers may distribute in low-density areas between true clusters. This is due to an excessive number of micro-cluster centers in conjunction with imbalanced data distribution, resulting in low-density areas between small clusters and real clusters being connected by redundant micro-cluster centers. These redundant micro-cluster centers, influenced by the network structure, cannot be fully distributed within the actual cluster regions. As a result, these micro-cluster centers fail to effectively represent a group of similar objects and may even be mistakenly identified as real clusters, resulting in unsatisfactory clustering accuracy.

Definition 1. *Given a micro-cluster center $\mathbf{z}_i \in \mathbb{R}^d$, if the local topological density of $\mathbf{z}_i$'s neighbor is larger than its own topological density, i.e. $r(\mathbf{z}_i) > 1$, then such center is called Bridge Node.*

We define topological density as:

$$d_s\left(\mathbf{z}_i\right) = \frac{\left|\psi\left(\mathbf{z}_i\right)\right|}{\sum_{\mathbf{z}_j \in \psi(\mathbf{z}_i)} \left\|\mathbf{z}_i - \mathbf{z}_j\right\|^2}, \tag{4}$$

$\left|\psi\left(\mathbf{z}_i\right)\right|$ refers to the number of the neighbors of $\mathbf{z}_i$.

Bridge Node (BN) acts as a bridge connecting two micro-clusters that should remain separate during the merging phase, which leads to a degradation in cluster performance. To identify BN, we introduce the local topological density ratio:

$$r\left(\mathbf{z}_i\right) = \frac{\sum_{\mathbf{z}_j \in \psi(\mathbf{z}_i)} d_s\left(\mathbf{z}_j\right)}{\left|\psi\left(\mathbf{z}_i\right)\right| \cdot d_s\left(\mathbf{z}_i\right)}, \tag{5}$$

when $r(\mathbf{z_i}) > 1$, $\mathbf{z}_i$ is identified to be BN. In the TDP stage, eliminating BNs could remove the micro-cluster centers in low-density regions between real clusters, leaving micro-cluster centers that better reflect the distribution of data points. This enhancement in the representation capability of micro-cluster centers allows data points to be assigned to their nearest micro-cluster centers.

C. TGM: Topology-Guided Merging

We then merge the micro-cluster updated after BN removal to obtain the final partition. Our merge process utilizes two metrics, namely compactness and separability. Micro-clusters with low separability should be merged first, as they are likely to belong to the same final cluster. For the separability between two micro-clusters C_i and C_j, it will be defined as follows:

$$\mathfrak{d}_{ij} = \frac{1}{\min\limits_{u \in U} S\left(u\right)}, \tag{6}$$

where

$$S\left(u\right) = \frac{\left|C_i\right| f\left(u|0.5, \sigma_i^2\right) + \left|C_j\right| f\left(u|-0.5, \sigma_j^2\right)}{\left|C_i\right| + \left|C_j\right|}, \tag{7}$$

$U = \{-0.5, -0.49, \ldots, 0.5\}$ and $S(u)$ is the 1-D Gaussian mixture probability density function. The probability density function of a Gaussian distribution $f(u|0.5, \sigma_i^2)$ describes data points in C_i. The data objects $\mathbf{x}$ within C_i and C_j should be projected onto the topological connection line as 1-D objects

$$\mathbf{x}' = \frac{(\mathbf{x} - \mathbf{z}_m)^T (\mathbf{z}_i - \mathbf{z}_j)}{\left\|\mathbf{z}_i - \mathbf{z}_j\right\|^2} \tag{8}$$

to obtain the data distribution, where $\mathbf{z}_m = \left(\mathbf{z}_i + \mathbf{z}_j\right)/2$, and $\mathbf{z}_i$ and $\mathbf{z}_j$ are the endpoints of the mapping connection line (at 0.5 and -0.5). In Eq. (7), σ^2 represents the variance of the distribution for the $\mathbf{x}'$.

With the increasing of the distance between two micro-clusters, $\min_{u \in U} S(u)$ is approaching 0, causing an increase in $\mathfrak{d}_{ij}$. We will use the separability metric between micro-clusters to identify those that should be merged. The final retained micro-clusters, which have not been merged, are the micro-cluster clustering results of the current data chunk. After merging the C_i and C_j, k micro-clusters are updated to $k-1$ micro-clusters. The computation can be stopped when $k = 1$.

The global compactness of the clustering result, represented by com, can be understood as the minimum separability value between adjacent micro-clusters. Therefore, a smaller com value indicates a better clustering. The following equation computes the global compactness:

$$com_{k-1} = \min_{1<i<k} \min_{\substack{\mathbf{z}_j \in \psi(\mathbf{z}_i) \\ j>i}} \mathfrak{d}_{ij}. \tag{9}$$

Another metric is the global separability. Before that, let's define the $\bar{\mathbf{z}}_g$ as the one with minimum local topological density:

$$\bar{\mathbf{z}}_g = \min_{\mathbf{z}_q \in Z} d_s\left(\mathbf{z}_q\right) \tag{10}$$

is the center of C_g, which is generally distributed in the outer regions of true clusters to represent low-density data. Therefore, we can conclude that:

$$sep_k = \sum_{\mathbf{x}_j \in C_g} \left(\frac{q_j}{\kappa}\right), \tag{11}$$

where κ refers to the number of nearest neighbors, and q_j represents the number of κ-nearest neighbors of $\mathbf{x}_j$ that do not belong to the C_g. An approximate search algorithm [20] is employed, to reduce execution time.

Clearly, a smaller sep value indicates a better clustering result, as each cluster exhibits a higher degree of separability. During the aggregation process, the com value changes in ascending order, and the sep value changes in descending order. Therefore, we can automatically determine the number of clusters as follows:

$$k^* = \underset{1<K<k-1}{\arg\min} \left(\frac{com_K}{\max\limits_{k'}\{com_{k'}\}} + \frac{sep_K}{\max\limits_{k'}\{sep_{k'}\}} \right). \tag{12}$$

To normalize the global compactness and separability, we achieve this by dividing by their maximum. Finally, we select the minimum value from the sum of global separability and compactness to guide the selection of the clustering result.

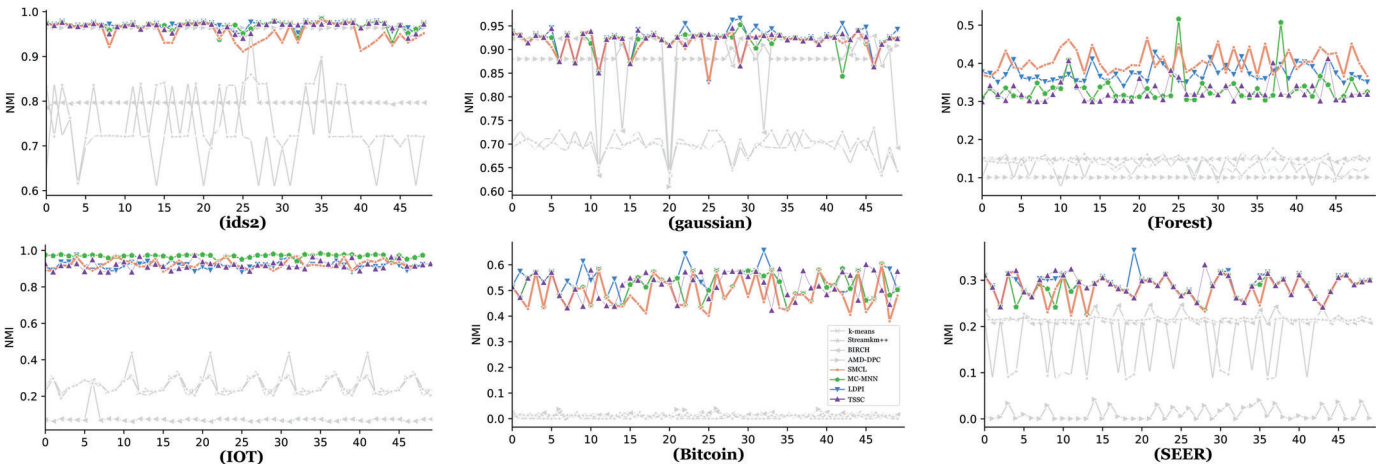

Fig. 2. Performance on each data chunk of the six large-scale IDD datasets.

D. TSSC: Complexity Analysis

To analyze the time complexity of the proposed algorithm, we assume that each incoming dataset is a data chunk $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, and the initial number of neurons is Q, where $Q \ll n$. In the TR, the time complexity for training RSOM and k-means is $O(Q * n)$. In the TDP stage, generating the adjacency micro-cluster centers matrix has a time complexity of $O\left(Q^2\right)$. In the TGM stage with $\hat{Q}$ micro-clusters, we iteratively calculate global separability and compactness degrees, resulting in a time complexity of $O\left(\frac{n}{\hat{Q}} \log \frac{n}{\hat{Q}}\right)$. In summary, the overall computational cost of TSSC is $O\left(Q * n + Q^2 + \frac{n}{\hat{Q}} \log \frac{n}{\hat{Q}}\right)$.

III. Experiments

A. Experiment Settings

Ten datasets[1] including two biomedical, four real benchmark, and four synthetic datasets are used for the experiments. Statistics of the datasets are combined with the comparative results in Table I. For the biomedical datasets, haberman [21] and wpbc [22] are related to breast cancer in the biomedical field. The haberman dataset contains cases from a study conducted on the survival of patients who underwent surgery for breast cancer, while the wpbc dataset includes cases that exhibited invasive breast cancer without evidence of distant metastasis at diagnosis. We chose them to validate the robustness and accuracy of our method on biomedical datasets of varying sizes. For the synthetic datasets, banana and lithuanian are generated using PRTools with thickness parameter set at 0.4 and choose 'non-spherical clusters'. ids2 and gaussian [13] are synthesized by applying a mixture of bivariate Gaussian density functions with custom scales in the order of one million, resulting in spherical clustering. Min-max normalization is adopted to pre-process each feature to be with the identical value domain $[0, 1]$.

We compare TSSC with seven other methods, namely, k-means [23], SKM++ [8], BIRCH [7], AMD-DPC [18], SMCL [5], MC-MNN [24], and LDPI [14]. k-means represents the traditional partition-based clustering approach, partition-based SKM++, hierarchy-based BIRCH, and density-based AMD-DPC are representative advanced algorithms specifically designed for drifting data. SMCL, MC-MNN, and LDPI are state-of-the-art methods suitable for static imbalance-drifting data. For k-means, SKM++, BIRCH, and AMD-DPC, since they require the specification of the number of clusters in advance, SMCL, MC-MNN, LDPI, and our method can adaptively determine the number of clusters without specification. According to the experimental results, the specific experimental parameters of our method are as follows: initial number of neurons $Q^2 = 100$, and the number of nearest neighbors $\kappa = 10$. The other parameters of all the compared methods are set according to the recommendation in the source literature.

Two metrics, i.e., Normalized Mutual Information (NMI) [25] and the Difference of Coefficient of Variation (DCV) [5] are utilized for evaluation. All the reported performance are averaged over ten runs with random initialization of seeds.

B. IDD-Gen Evaluation Augmentation

An IDD Generation (IDD-Gen) algorithm has been designed to generate streaming data chunks by performing two-layer randomization on cluster size and imbalance ratio based on given real datasets. Its pseudocode can be found in the Supplementary Material[2]. In the subsequent three sub-experiments, IDD-Gen is repeated N times to yield N data chunks on each dataset. With a large N, the data chunks will traverse various complex and challenging IDD situations beyond the realistic, i.e., extreme imbalance ratio and more frequent drifting for performance evaluations. In other words, IDD-Gen enhances the diversity of experimental datasets and enables the execution of more thorough and stringent validation procedures. Consequently, it allows for a comprehensive evaluation of clustering performance across various IDD scenarios.

[1] https://archive.ics.uci.edu/

[2] https://github.com/515150338/Detailed-information-about-BIBM2024

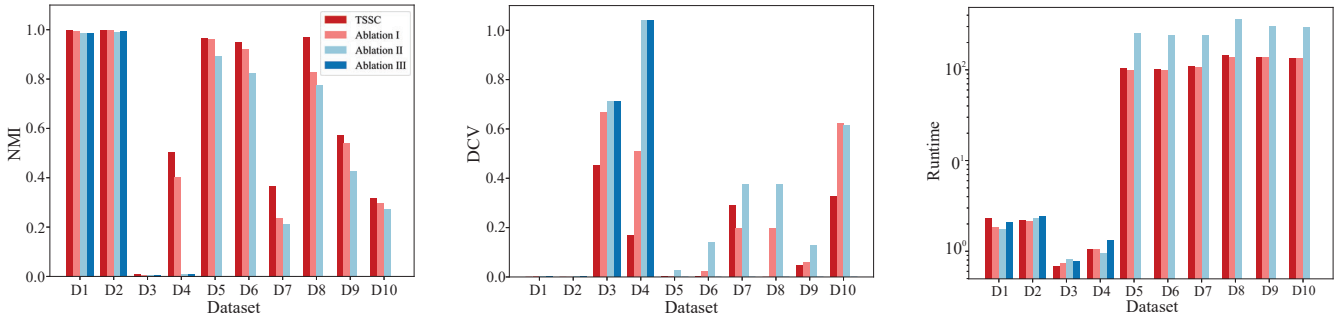

Fig. 3. Visualization of ablation effects on all the ten datasets sorted out in Table I.

TABLE I

STATISTICS OF THE TEN DATASETS AND CLUSTERING PERFORMANCE. THE BEST AND SECOND-BEST RESULTS ARE SHOWN IN BOLDFACE AND UNDERLINED, RESPECTIVELY. 'OOM' MEANS THAT THE METHODS CANNOT OUTPUT CLUSTERING RESULTS DUE TO 'OUT OF MEMORY'.

Dataset	Measures	k-means	SKM++	BIRCH	AMD-DPC	SMCL	MC-MNN	LDPI	TSSC (ours)	# Samples	# Clusters	IR
lithuanian	NMI	0.0009	0.0012	0.0452	0.0548	**1.0000**	**1.0000**	**1.0000**	**1.0000**	2.00K	2	5.00
	DCV	0.9815	0.7452	1.0228	0.9289	**0.0000**	**0.0000**	**0.0000**	**0.0000**			
banana	NMI	0.0943	0.9206	0.6512	0.2631	0.9745	**1.0000**	**1.0000**	**1.0000**	2.40K	2	5.00
	DCV	0.8221	0.7995	0.9195	0.4057	0.0791	**0.0000**	**0.0000**	**0.0000**			
haberman	NMI	0.0007	0.0015	0.0003	0.0469	0.0011	**0.1343**	0.1002	0.0088	0.31K	2	2.77
	DCV	1.0289	1.0160	1.3489	0.4612	0.4416	**0.3545**	0.3991	0.4529			
wpbc	NMI	0.6175	0.6231	0.3219	0.1329	0.6339	**0.6355**	0.4353	0.5052	0.57K	2	1.69
	DCV	1.1217	1.2710	1.2335	0.1870	0.1396	0.1564	**0.1028**	0.1693			
ids2	NMI	0.4312	0.5092	0.7966	0.9312	OOM	OOM	OOM	**0.9648**	1.00M	5	10.00
	DCV	0.5337	0.4766	0.7919	0.0312	OOM	OOM	OOM	**0.0023**			
gaussian	NMI	0.5022	0.5498	0.9234	0.9358	OOM	OOM	OOM	**0.9498**	1.00M	4	19.87
	DCV	0.6092	0.5720	0.5253	0.0406	OOM	OOM	OOM	**0.0031**			
Forest	NMI	0.1374	0.1459	0.1487	0.2648	OOM	OOM	OOM	**0.3677**	0.58M	7	103.13
	DCV	0.8561	0.8092	1.4571	0.4113	OOM	OOM	OOM	**0.2901**			
IOT	NMI	0.2011	0.2310	0.0757	0.7951	OOM	OOM	OOM	**0.9703**	2.00M	2	2.54
	DCV	0.5099	0.4971	1.2376	0.1463	OOM	OOM	OOM	**0.0002**			
Bitcoin	NMI	0.0138	0.0139	0.0201	0.0192	OOM	OOM	OOM	**0.5722**	1.00M	2	23.19
	DCV	1.2386	0.8639	0.8257	1.0395	OOM	OOM	OOM	**0.0482**			
SEER	NMI	0.2185	0.2202	0.2123	0.0402	OOM	OOM	OOM	**0.3155**	1.00M	3	3.99
	DCV	1.4544	1.4517	1.6613	2.1138	OOM	OOM	OOM	**0.3252**			

C. Clustering performance Evaluation

Three sub-experiments are conducted in this subsection.

Runtime impact of chunk size: We conducted a runtime-data size analysis on six large-scale datasets, TSSC exhibits a similar running time to that of fast clustering methods like k-means, SKM++, BIRCH, and AMD-DPC, thanks to their linear or $O(n \log n)$-level complexity. In contrast, SMCL, MC-MNN, and LDPI experience 'out of memory' due to their polynomial time complexity. Due to space limitations, we will only discuss the reasons behind these observations and will not provide the detailed results.

Clustering performance on entire dataset: The results reported in Table I illustrate that TSSC outperforms other methods, winning in 16 out of 20 comparisons. Compared to fast clustering methods like k-means, SKM++, BIRCH and AMD-DPC, TSSC excels in detecting imbalance-drifting clusters. When compared to SMCL, MC-MNN, and LDPI, TSSC continues to lead the way. SMCL, MC-MNN, and LDPI couldn't run on six large-scale datasets due to 'out of memory', while TSSC remains competitive on large-scale datasets.

IDD clustering performance on streaming data: For each of the six large-scale datasets, we generate 50 chunks by using IDD-Gen and let them flow in one by one per the time-stamp. The data volume of each chunk is determined based on the maximum data capacity that can be processed by SMCL. As seen in Fig. 2, performance on each data chunk of the large-scale IDD datasets, TSSC consistently outperforms the other four fast clustering methods. SMCL, MC-MNN, and LDPI are only feasible for individual chunks within large-scale datasets, reaffirming TSSC's competitiveness. In summary, the proposed TSSC is superior in both accuracy and scalability in handling imbalanced drifting data with different scales.

D. Ablation Study

Since TSSC comprises three components, we conducted module-specific ablations to analyze their impact on NMI, DCV, and Runtime. In Ablation I, we exclude the k-means fine-tuning step. Building upon the previous procedure, we omit the BN removal step to create Ablation II. Ablation III

use the SGMS merge method from SMCL, departing from TGM used in Ablation II. Specific ablation study results on the ten datasets are demonstrated in Fig. 3.

Through observing the experimental results, we discovered that ablating any module leads to decreased clustering performance, indicating that each module plays a necessary role in achieving good IDD clustering performance. Specifically, TR ablation has the least impact, while ablations of TDP and TGM result in more noticeable accuracy differences. When processing large-scale data, SGMS requires square-level running time, which can lead to 'out of memory' issues.

E. Parameter Analysis

In the proposed TSSC framework, there are two parameters, namely the number of neurons Q^2 and the neighbor count κ. Different values of these parameters can potentially affect the clustering performance in various ways. We conduct experiments on ten datasets by varying the values of Q and κ, recording the obtained NMI and DCV. Q values were traversed from 3 to 12 with a step size of 1, while κ values ranged from 3 to 21 with a step size of 2. TSSC framework demonstrates low sensitivity to parameter changes. Particularly, the κ value exhibits minimal influence on the clustering results. For appropriate Q values, small variations do not lead to a significant drop in precision, this also confirms the robustness of our framework in terms of hyper-parameters.

IV. Conclusion

This paper proposes a new clustering paradigm for imbalanced and drifting data called TSSC, which leverages SOM to achieve rapid and accurate clustering performance. Specifically, we fine-tune the nodes of the self-organized topology to effectively partition the dataset into compact micro-clusters. During the merging stage, we utilize topological information to swiftly retrieve similar clusters, thereby significantly reducing computation costs. To further validate our approach, IDD-Gen is designed to generate more challenging streaming IDD data. TSSC features efficiency, accuracy, and hyper-parameter robustness. More importantly, the trained SOM topology guides the whole data processing, making TSSC highly explainable for the understanding of complex biological, medical, and health big data. Extensive experiments including comparative evaluation, ablation studies, and parameter sensitivity analysis, have been conducted to demonstrate the efficacy of TSSC.

While TSSC proves effective, it is not exempt from limitations. A possible future direction is to further achieve linear time complexity in IDD clustering. Moreover, extending the current approaches for handling heterogeneous feature data would also be promising in enhancing their applicability to real-world datasets, particularly in offering powerful solutions to address the challenges posed by complex biomedical data.

References

[1] L. Zhao, Y. Zhang, X. Luo, Y. Zhang, Y.-M. Cheung, and K. Li, "Selecting heterogeneous features based on unified density-guided neighborhood relation for complex biomedical data analysis," in *BIBM*. IEEE, 2023, pp. 771–778.

[2] L. Zhao, Y. Zhang, Y. Ji, A. Zeng, F. Gu, and X. Luo, "Heterogeneous drift learning: classification of mix-attribute data with concept drifts," in *DSAA*. IEEE, 2022, pp. 1–10.

[3] M. Zhao, Y. Zhang, Y. Ji, and Y. Lu, "Unsupervised concept drift detection via imbalanced cluster discriminator learning," in *PRCV*. Springer, 2023, pp. 31–43.

[4] Y. Zhang and Y.-M. Cheung, "Graph-based dissimilarity measurement for cluster analysis of any-type-attributed data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 9, pp. 6530–6544, 2023.

[5] Y. Lu, Y.-M. Cheung, and Y. Y. Tang, "Self-adaptive multiprototype-based competitive learning approach: A k-means-type algorithm for imbalanced data clustering," *IEEE Transactions on Cybernetics*, vol. 51, no. 3, pp. 1598–1612, 2019.

[6] Y. Chen and L. Tu, "Density-based clustering for real-time stream data," in *ACM SIGKDD*, Aug 2007.

[7] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: A new data clustering algorithm and its applications," *Data Mining and Knowledge Discovery*, vol. 1, no. 2, pp. 141–182, Jan 1997.

[8] M. Ackermann, M. Märtens, C. Raupach *et al.*, "Streamkm++ a clustering algorithm for data streams," *Journal of Experimental Algorithmics (JEA)*, vol. 17, pp. 1–2, 2012.

[9] R. Liu, H. Wang, and X. Yu, "Shared-nearest-neighbor-based clustering by fast search and find of density peaks," *Information Sciences*, vol. 450, pp. 200–226, 2018.

[10] Y. Zhang, Y.-M. Cheung, and Y. Liu, "Quality preserved data summarization for fast hierarchical clustering," in *IJCNN*. IEEE, 2016, pp. 4139–4146.

[11] Y.-M. Cheung and Y. Zhang, "Fast and accurate hierarchical clustering based on growing multilayer topology training," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 3, pp. 876–890, 2018.

[12] M. Liu, X. Jiang, and A. C. Kot, "A multi-prototype clustering algorithm," *Pattern Recognition*, vol. 42, no. 5, pp. 689–698, May 2009.

[13] J. Liang, L. Bai, C. Dang, and F. Cao, "The *k*-means-type algorithms versus imbalanced data distributions," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 4, pp. 728–745, 2012.

[14] W. Tong, Y. Wang, and D. Liu, "An adaptive clustering algorithm based on local-density peaks for imbalanced data without parameters," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 3419–3432, 2023.

[15] D. Li, S. Zhou, T. Zeng, and R. H. Chan, "Multi-prototypes convex merging based k-means clustering algorithm," *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[16] F. Cao, M. Estert, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise." in *SDM*, Apr 2006.

[17] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[18] D. Amagata, "Scalable and accurate density-peaks clustering on fully dynamic data," in *Big Data Conference*. IEEE, 2022, pp. 445–454.

[19] N. Rougier and G. Detorakis, "Randomized self organizing map," *arXiv: Neural and Evolutionary Computing,arXiv: Neural and Evolutionary Computing*, Nov 2020.

[20] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 04, pp. 824–836, apr 2020.

[21] S. Haberman, "Haberman's Survival," UCI Machine Learning Repository, 1999, DOI: https://doi.org/10.24432/C5XK51.

[22] S. W. Wolberg, William and O. Mangasarian, "Breast Cancer Wisconsin (Prognostic)," UCI Machine Learning Repository, 1995, DOI: https://doi.org/10.24432/C5GK50.

[23] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[24] W. Tong, Y. Wang, D. Liu, and X. Guo, "A multi-center clustering algorithm based on mutual nearest neighbors for arbitrarily distributed data," *Integrated Computer-Aided Engineering*, vol. 29, no. 3, pp. 259–275, 2022.

[25] W. Rong, E. Zhuo, G. Tao, and H. Cai, "Effective and adaptive refined multi-metric similarity graph fusion for multi-view clustering," in *Advances in Knowledge Discovery and Data Mining*. Cham: Springer International Publishing, 2021, pp. 194–206.