TWStream: Three-Way Stream Clustering

Jiarui Sun, Mingjing Du, *Member, IEEE,* Zhenkang Lew, and Yongquan Dong

***Abstract*—A bunch of stream clustering algorithms have been proposed recently to mine data streams generated at high speeds from hardware platforms and software applications. Density-based methods are widely used because they can handle outliers and capture clusters of arbitrary shapes. However, it is still hard to effectively identify multi-density clusters with ambiguous boundaries in a data stream. To address these limitations, this paper introduces a data stream clustering algorithm called TWStream, based on the three-way decision theory. It is a two-stage clustering algorithm based on density. In the online stage, an augmented $k$nn graph is maintained incrementally to accelerate the update of the $k$nn graph. In the offline stage, TWStream introduces the concept of boundary confidence to detect cluster boundaries efficiently and reveal potential cores of clusters. It integrates the skewness and sparsity of the data distribution, as well as the evolving trend of the stream. In the next step, a micro-cluster-based three-way clustering strategy is applied to reconstruct latent clusters. It improves the clustering quality of boundary-ambiguous clusters in a stream using a mutual reachability-based clustering approach and a three-way assignment approach. The proposed algorithm is compared with 9 competitors on 15 data streams. Experimental results show TWStream achieves competitive performance, verifying its effectiveness. The source code of the proposed TWStream can be available at https://github.com/Du-Team/TWStream.**

***Index Terms*—Three-way decision, data stream, three-way clustering, density-based clustering, uncertain data analysis.**

I. Introduction

As hardware and software technology continues to develop, more and more application devices are used in a wide range of fields. Examples include network monitoring, satellite remote sensing, mobile communications, banking transactions, etc. These application devices interconnect and generate vast amounts of data at high speed. Data growing over time is known as a data stream. Data streams necessitate real-time online acquisition and processing to ensure rapid responsiveness due to their massive and unbounded nature. The processing of data streams in real-time has become a hot research area.

The key to understanding and utilizing streams is finding patterns hidden in them. Cluster analysis turns out to be the first tool for data stream mining since hand-labeling is expensive. Several efforts have been made in this direction [1], [2], [3].

In most data stream clustering algorithms, each data object is assigned to only one cluster. These hard clustering algorithms tend to result in higher error rates or decision risks when multi-density clusters with ambiguous boundaries are present in data streams. Instead of a two-way decision, a three-way decision [4] is based on a type of human thinking termed triadic thinking [5]. It divides a universal set into three disjoint regions and makes three types of decisions accordingly to achieve the desired result [6], [7].

According to the triadic thinking of the three-way decision, three-way clustering [8] defines three statuses between objects and clusters: belong-to, ambiguity and not belong-to. Based on these statuses, a three-way cluster is represented by a pair of nested sets called lower bound and upper bound, respectively. Thus, the entire discussion region is divided into three parts: positive region (*POS*), boundary region (*BND*), and negative region (*NEG*). Simple examples of two-way and three-way cluster representations are shown in Fig. 1 [9]. The objects in the positive region are definitely part of the cluster. Objects in the boundary region may be part of the cluster but may also belong to other clusters. The objects in the negative region are not likely to belong to the cluster.

Existing density-based stream clustering algorithms focus either on clustering multi-density clusters (e.g., MR-Stream [10] and MuDi-Stream [11]) or resolving boundary-ambiguous clusters (e.g., DBSTREAM [12] and CEDAS [13]). To our knowledge, however, none of the existing stream clustering algorithms can address both problems simultaneously. To address the aforementioned problems, this paper proposes a data stream clustering algorithm using the three-way decision theory called TWStream (**T**hree-**W**ay **Stream** Clustering). We introduce a notion of boundary confidence based on spatial vector decomposition to classify core and boundary micro-clusters. It integrates the skewness and sparsity of the data distribution, as well as the evolving trend of the stream. In addition, we introduce the three-way decision theory to reconstruct cluster core and boundary regions. To our knowledge, the algorithm proposed in this paper represents the inaugural effort to integrate the three-way decision theory [4] into data stream clustering. The information processing paradigm based on the three-way decision theory further improves the accuracy of data assignment. Experimental results on various synthetic and real-world data streams show that TWStream can effectively solve the above problems. The main contributions of this work can be summarized as follows:

- A density-based two-stage stream clustering algorithm is proposed. In the online stage, an augmented $k$nn graph is kept in memory. In the offline stage, it performs clustering based on boundary confidence and three-way decision theory, thereby improving the quality of clustering.
- The concept of boundary confidence is proposed to detect cluster boundaries efficiently and reveal potential cores of clusters. It integrates the skewness and sparsity of the data distribution, as well as the evolving trend of the stream.

This work was supported in part by the National Natural Science Foundation of China under Grant 62006104 and Grant 61872168. *(Corresponding author: Mingjing Du.)*

J. Sun, M. Du, Z. Lew, and Y. Dong are with the School of Computer Science and Technology, Jiangsu Normal University, Xuzhou 221116, China. (e-mail: sunjr@jsnu.edu.cn; dumj@jsnu.edu.cn; 7547461@gmail.com; tomdyq@163.com).

- A micro-cluster-based three-way clustering strategy is proposed to reconstruct potential clusters effectively, which improves the clustering quality of boundary-ambiguous clusters in a stream.

The rest of this paper is organized as follows: Sec. II reviews studies related to our work. The relevant preliminary and definitions are introduced in Sec. III. The critical components and techniques of the proposed algorithm are explained in detail in Sec. IV. Sec. V presents experimental results to evaluate the algorithm's modelling capabilities on synthetic and real-world datasets. Finally, concluding remarks are drawn in Sec. VI.

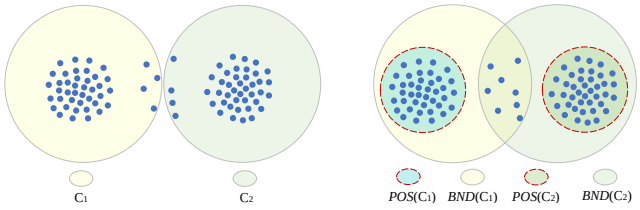


Fig. 1: Cluster representations of (a) two-way clustering and (b) three-way clustering.

II. Related Work

In this section, we review the main density-based stream clustering algorithms, as well as the three-way clustering approaches [14] most closely related to our study.

A. Online-offline Paradigm

Aggarwal *et al.* [15] propose the first two-stage evolving data stream clustering framework, CluStream. CluStream is based on a *pyramid time model* that enables users to obtain clustering results at various time horizons. CluStream includes the concept of micro-cluster, which is extended by the *Clustering Feature Vector* (CF) [16]. As a result, less memory is required, and computational efficiency is improved. The offline stage of CluStream is based on a variant of k-means for clustering, which leads to two main drawbacks: the number of clusters needs to be specified in advance, and only spherical clusters can be captured. DenStream [17] applies the two-stage framework to a density-based clustering algorithm. It is capable of identifying clusters of arbitrary shapes and cluster numbers need not be predefined. Furthermore, it employs a damped window model to capture the evolution of a data stream. In the offline stage, a variant of the DBSCAN algorithm [18] is applied to determine the density connectivity between core micro-clusters to form the final macro-clusters. HDDStream [19] addresses the clustering problem for high-dimensional streaming data using a density-based projection strategy. A recently proposed clustering algorithm based on a shared density graph between micro-clusters is named DBSTREAM [12]. Connectivity between micro-clusters depends on the density of shared areas. It is capable of identifying low-density regions between adjacent micro-clusters, thus avoiding erroneous merging operations.

The grid structure is another method of summarizing data streams. A representative case of grid-based solutions is D-Stream [20]. It discretises the object space in each dimension to form a grid structure. The grid cells are populated in the online stage, and the final clusters are obtained by merging neighbouring high-density grids in the offline stage. A further extension [21] introduces the concept of *attraction* to deal with the issue of low-density regions between adjacent grid cells. In MR-Stream [10], all grid cells are recursively halved in each dimension to generate a hierarchical tree that enables the identification of clusters at multiple resolutions, thus improving grid-based clustering accuracy. MuDi-Stream [11] utilizes a hybrid approach based on both grids and micro-clusters to summarize information in the data stream. Grids are used as outlier buffers and to deal with multiple-density data. The reclustering operation no longer depends on a constant threshold ϵ but on the notion of local cluster density. However, the majority of the algorithms above merge adjacent micro-clusters (or grids) by setting a fixed threshold, which lacks dynamic adaptability and fails to identify clusters with varying densities.

B. Fully Online Paradigm

CEDAS [13] and EDMStream [22] are two recently proposed online stream clustering algorithms based on micro-cluster summary structures. CEDAS introduces a time-decaying *energy* property for micro-clusters to capture data stream evolution. In addition, CEDAS dynamically organizes micro-clusters as a graph to facilitate fast response (a connected component is considered a cluster). In contrast to the graph employed in CEDAS, EDMStream tracks active cluster-cells in real-time using a dependency tree (i.e., DP-Tree). A cluster is considered for each subtree resulting from pruning the DP-Tree. Essentially, EDMStream is an online version of Density Peak Clustering (DPC) [23]. However, CEDAS and EDMStream cannot be applied to high-dimensional data streams. To address this challenge, Li *et al.* [24] propose the well-known ESA-Stream, an efficient grid-based stream clustering algorithm. The algorithm introduces an efficient dimensionality reduction technique based on the grid density centroid and develops a parameter adaptive component. The excellent performance of ESA-Stream in clustering high-dimensional data streams makes it a vital work in the field of data stream clustering. However, the above algorithms ignore clustering scenarios for data streams with multi-density clusters or boundary-ambiguous clusters.

C. Three-way Approaches

Yu *et al.* [14] propose three-way clustering by introducing the three-way decision theory into clustering. This field has developed rapidly over the past few years, with many three-way clustering algorithms being developed. 3W-DPET [25] is a three-way clustering approach that combines evidence theory and density peak clustering. Unlike 3W-DPET, Wang and Yao [26] propose a contraction-and-expansion based three-way clustering framework called CE3. It can construct three-way clusters on any hard clustering result by using a contraction

operation to shrink a cluster to get the core region and then using an expansion operation to enlarge a cluster to get the boundary region. However, CE3 and 3W-DPET must specify the number of clusters in advance. As a result, such algorithms are subject to the weakness of relying heavily on prior knowledge.

To overcome the above shortcomings, several density-based three-way clustering approaches are proposed. Similar to 3W-DPET, 3W-ADPC [27] is also a three-way clustering approach based on density peaks. It introduces the concept of natural nearest neighbors to overcome the difficulty of tuning parameters. However, 3W-DPET and 3W-ADPC have difficulty dealing with cluster overlap. To address this challenge, Sun *et al.* [28] propose TW-RDPC, which introduces a boundary peeling strategy. Unlike 3W-DPET, 3W-ADPC, and TW-RDPC, Yu *et al.* [29] propose a three-way clustering approach (3W-DBSCAN) based on an improved DBSCAN. However, all the three-way clustering approaches mentioned above are less fault-tolerant. Du *et al.* [9] propose a multistep three-way clustering approach called M3W. By introducing the sequential three-way decision theory [30], more available information is obtained, which enhances the accuracy of clustering.

Additionally, various ensemble algorithms based on three-way clustering are proposed to improve clustering performance further. Jiang *et al.* [31] propose a shadowed set-based multi-granular three-way clustering ensemble called S-M3WCE. It incorporates shadowed sets and the idea of multi-granularity rough sets to enhance clustering quality. Moreover, Fan *et al.* [32] present a three-way density-sensitive spectral clustering approach to overcome the limitation of spectral clustering in handling multi-scale data. The approach introduces density-sensitive distance and three-way decision theory. To further improve clustering accuracy, an improved ensemble three-way spectral clustering algorithm is developed based on an ensemble strategy.

Although the above approaches improve the clustering quality of boundary ambiguous clusters, they cannot be directly adapted to a data stream.

III. Preliminary and Definitions

A. Data Stream and Micro-cluster-based Definitions

A data stream is a sequence of data that is massive and potentially unbounded. It is essential to note that data objects in the stream arrive at a high speed and can only be scanned once. A data stream can be represented formally as $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$, where $x_i \in \mathbb{R}^d$, $1 \leq d \in \mathbb{Z}^+$, $1 \leq i \leq n$, and $n \to \infty$. The characteristics of the data stream indicate the clustering process will encounter the following challenges [33], [34], [35], [36]:

1) Data streams cannot be stored in their entirety. Therefore, the clustering algorithm must express them in a compact form.
2) Each data object in the stream can only be read once and needs to be processed quickly.
3) Concept drift is the inherent characteristic of an evolving data stream. The clustering algorithm must accurately capture concept drift in the stream to obtain more precise clustering results.
4) To achieve improved clustering results, the clustering algorithm should promptly identify and interpret outliers and eliminate them whenever possible.

In response to the third challenge above, implementing a forgetting mechanism can effectively capture concept drifts. Typically, a forgetting mechanism is introduced by placing data objects from a stream into a specified time window model. There are three well-known window models: landmark window, sliding window, and damped window. Similar to DenStream [17], we consider the problem of clustering a data stream in the damped window model. We assign a weight to each data object in the data stream that decreases exponentially with time t. The weight of the data object x_i at time t is defined as follows:

$$W(x_i, t) = 2^{-\lambda(t-t_o)} \qquad (t > t_o) \tag{1}$$

where t_o is the arrival time of data object x_i. λ ($\lambda > 0$) is the *decay factor*. The higher the λ value, the lower the importance of historical data compared to more recent data.

In response to the first challenge above, it is clearly unrealistic to provide a precise result in a streaming environment. Therefore, we resort to an approximate result and introduce a summary representation called micro-cluster. The weight of a micro-cluster mc_i at time t is defined as follows:

$$W(mc_i, t) = \sum_{x_i \in V(mc_i, t)} W(x_i, t) \tag{2}$$

where $V(mc_i, t)$ is the set of data objects that are summarized into the mc_i at or before t.

Fig. 2 illustrates the micro-cluster structure designed in this paper as two regions: kernel region ($\leqslant r/2$) and shell region ($> r/2$). As data continuously arrive and evolve, the weight of a micro-cluster is constantly changing. If a new data object x_i is summarized into the nearest micro-cluster mc_i at time t, we incrementally update the weight of mc_i as follows for quickly calculating:

$$f(mc_i, x_i) = \begin{cases} 1, & d \leq \frac{r}{2} \\ exp(-(d - \frac{r}{2})^2/(2\sigma^2)), & \frac{r}{2} < d \leq r \end{cases} \tag{3}$$

$$W(mc_i, t) = 2^{-\lambda(t-t_o)} W(mc_i, t_o) + f(mc_i, x_i) \tag{4}$$

where t_o is the last moment before t. $d = \|x_i - c(mc_i, t_o)\|_2$, $c(mc_i, t_o)$ is the center of mc_i at time t_o. Following the choice of σ in [12] we set $\sigma = r/6$.

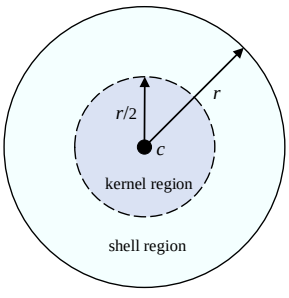


Fig. 2: The micro-cluster structure.

If x_i falls into the kernel region (i.e., $d \leq r/2$), the center of mc_i needs to be updated as follows:

$$c(mc_i, t) = (W(mc_i, t_o) \cdot c(mc_i, t_o) + x_i)/W(mc_i, t). \quad (5)$$

In response to the fourth challenge above, outliers in the data stream should be detected and handled promptly. In an evolving data stream, the role of clusters and outliers often exchange. Therefore, we establish an outlier weight threshold to determine whether a micro-cluster is active or not. The overall weight of a data stream is a constant $v/(1-2^{-\lambda})$, where v denotes the speed of stream, i.e., the number of objects arrived in one unit time [17]. Based on this knowledge, the outlier weight threshold is defined as follows [22]:

$$W_{min} = \frac{\beta v}{1-2^{-\lambda}} \quad (6)$$

where $\beta \in (1-2^{-\lambda}, 1)$ is the *weight threshold factor*. The larger the β value, the less the number of active micro-clusters.

Thus, the status of a micro-cluster mc_i at time t can be determined as follows:

$$status = \begin{cases} active, & \text{if } W(mc_i, t) \geq W_{min} \\ inactive, & \text{otherwise.} \end{cases} \quad (7)$$

Let $\mathbf{V}$ be a set including all active micro-clusters in memory at time t. All inactive micro-clusters are cached in the *Outlier Pool* (see IV-A).

B. Clustering Representation by Three-way Decision

Let $\mathbf{C} = \{C_1, C_2, \ldots, C_K\}$ be a family of clusters, and $\mathbf{U} = \{x_1, x_2, \ldots, x_n\}$ be a finite universe of objects that is not empty. In three-way clustering, a cluster C_i is depicted by a pair of nested sets [37]:

$$C_i = [\underline{C_i}, \overline{C_i}] \quad (8)$$

where $\underline{C_i}$ is the lower bound of C_i and $\overline{C_i}$ is the upper bound of C_i, $\underline{C_i} \subseteq \overline{C_i} \subseteq \mathbf{U}$.

Using *POS*, *BND*, and *NEG* to denote the three regions of cluster C_i respectively, the three-way clustering representation is as follows:

$$\begin{aligned} POS(C_i) &= \underline{C_i} \\ BND(C_i) &= \overline{C_i} - \underline{C_i} \\ NEG(C_i) &= \mathbf{U} - \overline{C_i} \end{aligned} \quad (9)$$

Objects in $POS(C_i)$ definitely belong to C_i. Objects in $BND(C_i)$ may belong to C_i or other clusters, and more information is needed to make judgements. Objects in $NEG(C_i)$ definitely do not belong to C_i.

Therefore, the three-way clustering results can be described by interval sets as follows:

$$\mathbf{C} = \{[\underline{C_1}, \overline{C_1}], [\underline{C_2}, \overline{C_2}], \ldots, [\underline{C_K}, \overline{C_K}]\}. \quad (10)$$

As a general rule, the subsets of a three-way cluster must satisfy the following conditions:

1) Non-Emptiness: $POS(C_i) \neq \emptyset$
2) Mutual Exclusion:
   $POS(C_i) \cap BND(C_i) = \emptyset$;
   $BND(C_i) \cap NEG(C_i) = \emptyset$;
   $POS(C_i) \cap POS(C_j) = \emptyset, i \neq j$
3) Complementarity: $POS(C_i) \cup BND(C_i) \cup NEG(C_i) = \mathbf{U}$

IV. Proposed Algorithm

In this section, we first introduce the overall framework of the proposed algorithm. Next, the key components and techniques of the proposed algorithm are explained in detail. Finally, we provide the algorithm pseudo-code along with the complexity analysis.

A. Overview

The proposed TWStream framework is outlined in Fig. 3. TWStream consists of two stages with a total of five components that collaborate with each other to cluster data streams efficiently. The functions of these five components are described as follows:

- **Data Stream Absorber:** It is responsible for receiving data objects from a stream and summarizing them into the appropriate micro-clusters (see Sec. III-A for details).
- **Outlier Pool:** It caches micro-clusters (inactive status) with weights below W_{min} (see Eq. (6)), which may be reactivated in the future.
- **Graph Manager:** It maintains an *augmented knn graph* incrementally to accelerate the update of the $k$nn graph (see Sec. IV-B for details).
- **Confidence Detector:** It detects cluster boundaries efficiently and reveals potential cores of clusters in a stream environment (see Sec. IV-C for details).
- **Three-Way Clustering Engine:** It reconstructs potential clusters employing a micro-cluster-based three-way clustering strategy effectively (see Sec. IV-D for details).

B. Online Graph Management

Fig. 3 illustrates that the primary function of the online stage is to efficiently maintain the k-nearest-neighbor information for each active micro-cluster through the *Graph Manager*. We generalize the $k$nn graph to the data stream setting, where each vertex in the graph is an active micro-cluster. To facilitate the following description, we first define the distance between micro-clusters.

Definition 1 (*Distance Between Micro-clusters, δ*)**.** Let mc_i and mc_j be micro-clusters. $c(mc_i, t)$ and $c(mc_j, t)$ are the centers of mc_i and mc_j, respectively. The distance between mc_i and mc_j is defined as:

$$\delta(mc_i, mc_j) = \|c(mc_i, t) - c(mc_j, t)\|_2 . \quad (11)$$

Then, the definition of k nearest neighbor micro-clusters can be given as follows:

Definition 2 (k *Nearest Neighbor Micro-clusters, N_k*)**.** Let mc_l be the kth nearest neighbor of micro-cluster mc_i at time t. The k nearest neighbors of micro-cluster mc_i at time t are a set of active micro-clusters mc_j with $\delta(mc_i, mc_j) \leq \delta(mc_i, mc_l)$, i.e., $N_k(mc_i, t) = \{mc_j | W(mc_j, t) \geq W_{min}, \delta(mc_i, mc_j) \leq \delta(mc_i, mc_l)\}$.

Furthermore, utilizing the $k$nn graph can obtain the mutual k-nearest neighbors of any micro-cluster, which is crucial for implementing the following technique (see Definition 6). The

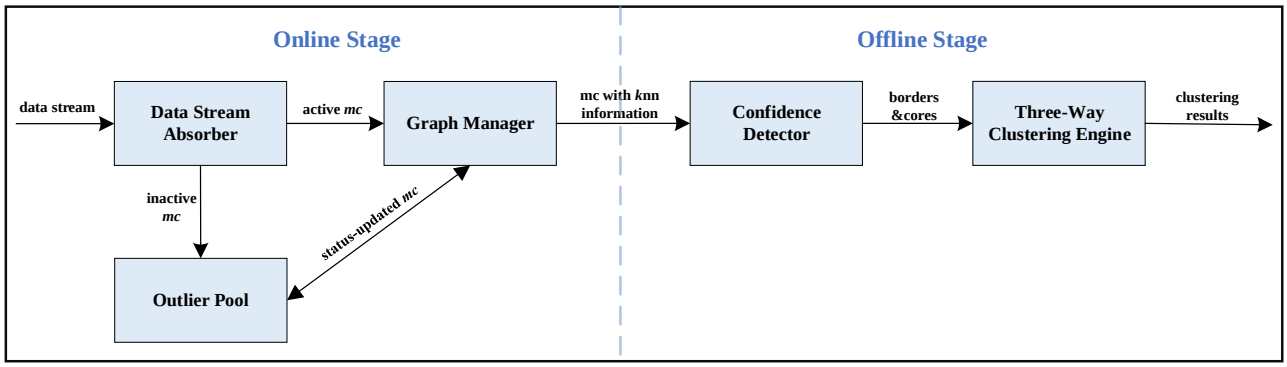


Fig. 3: The framework of TWStream. Online stage components: Data Stream Absorber, Outlier Pool, and Graph Manager. Offline stage components: Confidence Detector and Three-Way Clustering Engine. The three components in the online stage run continuously, whereas the two components in the offline stage are activated solely upon receiving a clustering request.

definition of mutual k-nearest neighbor micro-clusters can be given as follows:

Definition 3 (*Mutual k-Nearest Neighbor Micro-clusters, MN_k*)**.** The mutual k-nearest neighbors of micro-cluster mc_i at time t are a set of active micro-clusters mc_j that consider mc_i as its one of the k nearest neighbors and mc_j as mc_i one of the k nearest neighbors, i.e., $MN_k(mc_i, t) = \{mc_j | mc_j \in N_k(mc_i, t), mc_i \in N_k(mc_j, t)\}$.

Due to page limitations, we provide a detailed description of the *Graph Manager* in the Supplementary Material.

C. Boundary Confidence Detection

In recent years, Border-Peeling Clustering [38] has been proposed as a new clustering paradigm. It peels off the outer boundary objects iteratively to reveal the core regions of clusters. However, the iterative peeling process is inefficient and cannot be applied to data streams. Inspired by CDC [39], this paper proposes a boundary object detection method without iteration in the data stream environment. Based on the original data space and projected subspaces [40], this method fully considers the skewness and sparsity of the data. It can reveal the potential core regions of clusters effectively and efficiently. Using this method, the *Confidence Detector* component supports the subsequent *Three-Way Clustering Engine*.

In this study, we define a micro-cluster's neighborhood as its k nearest neighbors. As shown in Fig. 4(a), it can be observed that the neighbourhood distribution of the core micro-clusters located in the interior is relatively uniform and dense. Conversely, the neighbourhood distribution of boundary micro-clusters in the exterior is skewed and sparse. To quantify such distributional characteristics, we propose a notion of direction dispersion in the original data space.

Definition 4 (*Direction Dispersion, η*)**.** Let mc_i be an active micro-cluster at time t. With mc_i as the origin, the data space is divided equally into 2^d quadrants along d dimensions. $\mathbf{N} = \{n_1, n_2, \ldots, n_{2^d}\}$ is the number of the k nearest neighbors of mc_i within each quadrant. The direction dispersion of mc_i is defined as:

$$\eta_i = \frac{1}{k} \sum_{p} (n_p - \frac{k}{2^d})^2 \tag{12}$$

where $n_p \in \mathbf{N}$ and $n_p > 0$.

The direction dispersion reflects whether the k nearest neighbors of the micro-cluster mc_i within as many quadrants as possible. Furthermore, it reflects the degree of variation in the number of micro-clusters within these quadrants. A larger value of η_i indicates a more skewed distribution of data in the neighbourhood of the micro-cluster mc_i. It implies that mc_i is more likely in boundary regions. As shown in Figs. 4(c)-(d), the k nearest neighbors of the boundary micro-clusters mc_2 and mc_3 are primarily concentrated within a few quadrants. In contrast, Fig. 4(b) illustrates that the k nearest neighbors of the core micro-cluster mc_1 are uniformly distributed within four quadrants.

We project the micro-clusters in the neighbourhood along each dimension to further capture the skewness characteristics of data distribution at a fine-grained level. The original data space is transformed into projection subspaces. According to the previous analysis, the k nearest neighbors of the core micro-cluster are distributed relatively uniformly around it. As shown in Fig. 4(b), the projections of the k nearest neighbors of the core micro-cluster mc_1 show a symmetric distribution around mc_1. In contrast, the projections of the k nearest neighbors of the boundary micro-cluster mc_2 show a skewed distribution around mc_2, as depicted in Fig. 4(c). Based on the above observations, we propose a notion of dimension skewness.

Definition 5 (*Dimension Skewness, γ*)**.** Let mc_i be an active micro-cluster at time t. The dimension skewness of mc_i is defined as:

$$\gamma_i = \sum_{h} \left| \sum_{j} mc_{ih} - mc_{jh} \right| \tag{13}$$

where $1 \leq h \leq d$ and $mc_j \in N_k(mc_i, t)$.

Obviously, the dimension skewness is an accumulation of each dimension's skewness. A larger value of γ_i indicates a more skewed distribution of data in the neighbourhood of the micro-cluster mc_i. It implies that mc_i is more likely in boundary regions (shown in Fig. 4(c)). Conversely, if the value of γ_i is smaller, the mc_i is likely to be a core micro-cluster (shown in Fig. 4(b)) or a boundary micro-cluster (shown in Fig. 4(d)).

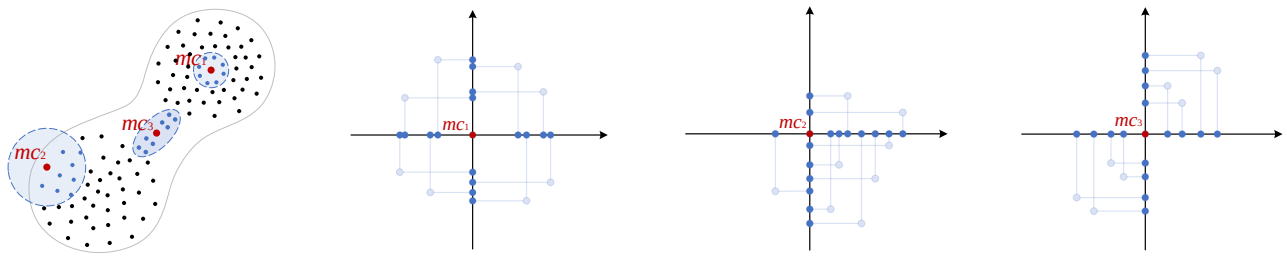


Fig. 4: An example of skewness analysis. (a) The data distribution, where black points represent micro-cluster centers. (b) The projection subspace of the k nearest neighbors of the core micro-cluster mc_1. (c) The projection subspace of the k nearest neighbors of the boundary micro-cluster mc_2. (d) The projection subspace of the k nearest neighbors of the boundary micro-cluster mc_3.

Further, we need to capture the sparsity of micro-cluster distribution to distinguish core micro-clusters from boundary micro-clusters better. The kernel density estimation (KDE) is an established method of measuring density. In terms of the kernel function, we choose the widely used Gaussian kernel function, which is smooth. In addition, the bandwidth is designed as an adaptive value [41] (i.e., the distance between the sample under test and its kth nearest neighbor) to detect clusters with varying densities in a data stream. Using the scheme, the bandwidth varies with the sample location. Based on the above knowledge, we propose a decay-based kernel density estimation.

Definition 6 (*Decay-based Kernel Density, ρ*)**.** Let mc_i be an active micro-cluster at time t. The decay-based kernel density of mc_i is defined as:

$$\rho_i = W(mc_i, t) \sum_j exp\left(\frac{-\delta(mc_i, mc_j)^2}{\epsilon_j^2}\right) \tag{14}$$

where $mc_j \in MN_k(mc_i, t)$, ϵ_j is the distance between mc_j and its kth nearest neighbor.

The decay-based kernel density includes information on the temporal and spatial distribution of micro-clusters. A high ρ value indicates a micro-cluster characterized by high freshness and dense distribution. Thus, it is capable of identifying the evolving trends of streams and discovering clusters with varying densities.

In summary, the boundary confidence of a micro-cluster is derived as follows.

Definition 7 (*Boundary Confidence, ϕ*)**.** Let mc_i be an active micro-cluster at time t. η_i, γ_i, and ρ_i are defined as above. The boundary confidence of mc_i is defined as:

$$\phi_i = \eta_i \cdot \gamma_i / \rho_i. \tag{15}$$

The boundary confidence indicates the degree to which a micro-cluster belongs to a cluster boundary. It contains the characteristics of data distribution captured from three perspectives [42]. The larger the ϕ_i value, the more likely micro-cluster mc_i is at a cluster boundary.

Based on the above definitions (i.e., Definitions 4-7), the potential core region of a cluster can be revealed by peeling off the boundary micro-clusters with high boundary confidence values. Let $\mathbf{V} = \{mc_1, mc_2, \ldots, mc_s\}$ be a set including all active micro-clusters in memory at time t. The peeling process involves three steps. The first step is to compute the boundary confidence of each micro-cluster by executing Eq. (15). The second step is to obtain a list L by sorting the boundary confidence values of all micro-clusters in descending order. The last step is to peel off the micro-clusters considered as boundaries (micro-clusters with high ϕ values) from $\mathbf{V}$. Formally, the set of peeled micro-clusters whose boundary confidence values are greater than a specified value is as follows:

$$\mathbf{V}_{border} = \{mc_i | mc_i \in \mathbf{V}, \phi_i \geq L[\lfloor s \times \tau \rfloor]\} \tag{16}$$

where s is the number of micro-clusters in the set $\mathbf{V}$. $\tau \in [0, 1]$ is the *boundary percentage factor*. It denotes the percentage of boundary micro-clusters among all active micro-clusters. A larger τ value indicates more boundary objects in the data stream. Then, the set of core micro-clusters is derived as follows:

$$\mathbf{V}_{core} = \mathbf{V} \setminus \mathbf{V}_{border}. \tag{17}$$

With the aid of the set $\mathbf{V}_{core}$ and the set $\mathbf{V}_{border}$ obtained from the above peeling operation, we reconstruct the three-way cluster structure efficiently by applying the technique described below.

D. Micro-cluster-based Three-Way Clustering (MC-TWC)

To improve the clustering quality of clusters with ambiguous boundaries, we propose a micro-cluster-based three-way clustering strategy to reconstruct latent clusters. The strategy is organised in two phases: The first phase is to cluster the micro-clusters in the set $\mathbf{V}_{core}$ using a mutual reachability-based approach to obtain the initial core regions of the three-way clusters. The second phase is to assign the micro-clusters in the set $\mathbf{V}_{border}$ using a three-way approach to obtain the final core and boundary regions, completing the reconstruction of the three-way clusters.

1) Mutual Reachability-based Clustering Approach: Inspired by HDBSCAN [43] and BP [38], we define a mutual reachability-based approach for clustering micro-clusters in the set $\mathbf{V}_{core}$. Firstly, we define mutual reachability as follows:

Definition 8 (*Mutual Reachability*)**.** Two micro-clusters mc_i, $mc_j \in \mathbf{V}_{core}$ are mutually reachable at time t, if there is

a series of micro-clusters (mc_{p_1},...,mc_{p_n}) with $p_1 = i$ and $p_n = j$, such that for every two adjacent micro-clusters mc_{p_m}, $mc_{p_{m+1}}$ the relation $mc_{p_{m+1}} \in MN_k(mc_{p_m}, t)$ holds.

Next, the mutually reachable micro-clusters in the set $\mathbf{V}_{core}$ are iteratively merged, resulting in the initial core regions of the three-way clusters. The clustering result can be denoted by $\mathbf{C} = \{C_1, C_2, \ldots, C_K\} = \{[\underline{C_1}, \overline{C_1}], [\underline{C_2}, \overline{C_2}], \ldots, [\underline{C_K}, \overline{C_K}]\}$. Since the boundary regions of the clusters have not yet been constructed, the clusters can be directly denoted by the core regions as $\mathbf{C} = \{\underline{C_1}, \underline{C_2}, \ldots, \underline{C_K}\}$.

2) Three-Way Assignment Approach: After establishing the initial core regions of the clusters, we use the three-way approach to assign the micro-clusters in the set $\mathbf{V}_{border}$. The assignment scheme is divided into two steps to reconstruct three-way clusters gradually.

(i). Initial Assignment: The micro-clusters in the set $\mathbf{V}_{border}$ are attempted to be assigned to the initial core or boundary regions of the corresponding clusters to obtain the final core regions and the initial boundary regions.

(ii). Further Assignment: The unallocated micro-clusters in the set $\mathbf{V}_{border}$ are assigned to the initial boundary regions of the corresponding clusters to obtain the final boundary regions.

Determine whether a micro-cluster belongs to the positive region (core region), boundary region, or negative region of a cluster based on its probability of being a member of that cluster. At time t, the probability that micro-cluster mc_i is a member of cluster C_r can be calculated as follows:

$$P(C_r|mc_i) = \frac{\left|N_k(mc_i, t) \cap \{mc_j | mc_j \in \underline{C_r}\}\right|}{|N_k(mc_i, t)|} \tag{18}$$

where $|\cdot|$ denotes the cardinality of a set.

According to the above equation, $P(C_r|mc_i)$ gives the percentage of micro-clusters in the core region of the cluster C_r among the k nearest neighbors of the micro-cluster mc_i. Furthermore, each micro-cluster $mc_i \in \mathbf{V}_{border}$ to be assigned has a probability distribution $P(\mathbf{C}|mc_i) = \{P(C_1|mc_i), \ldots, P(C_r|mc_i), \ldots, P(C_K|mc_i)\}$. The length K is the number of clusters. Next, we define a two-step assignment scheme based on the probability distribution. The details are given below.

Initial Assignment: The prerequisite is the existence of core micro-clusters in the k nearest neighbors of the micro-cluster mc_i at time t. i.e., $N_k(mc_i, t) \cap \mathbf{V}_{core} \neq \emptyset$.

If the core micro-clusters in the k nearest neighbors of mc_i belong only to the core region of the cluster C_r, i.e., if $|\{C_s | P(C_s|mc_i) > 0, C_s \in \mathbf{C}\}| = 1$ and $P(C_r|mc_i) > 0$, further judgement is required based on the probability value. If the probability value is not less than $1/K$, i.e., if $P(C_r|mc_i) \geq 1/K$, mc_i is assigned to the core region of C_r. Otherwise, mc_i is assigned to the boundary region of C_r.

If the core micro-clusters in the k nearest neighbors of mc_i do not all belong to the core region of the cluster C_r and its probability value of being a member of C_r is maximum, i.e., if $|\{C_s | P(C_s|mc_i) > 0, C_s \in \mathbf{C}\}| > 1$ and $P(C_r|mc_i) = max(P(\mathbf{C}|mc_i))$, further judgements need to be made based on comparisons between these probability values. Assume that there is a cluster C_p ($p \neq r$, $P(C_p|mc_i) > 0$), such that the difference value between the probability of mc_i being a member of clusters C_r and C_p is less than $1/K$, i.e., if $P(C_r|mc_i) - P(C_p|mc_i) < 1/K$, mc_i is assigned to the boundary regions of C_r and C_p. Otherwise, mc_i is assigned to the core region of C_r.

Further Assignment: The prerequisite is the absence of core micro-clusters in the k nearest neighbors of the micro-cluster mc_i at time t. i.e., $N_k(mc_i, t) \cap \mathbf{V}_{core} = \emptyset$.

If the nearest micro-cluster to mc_i in the final core regions belongs to the cluster C_r, i.e., if $d(mc_i, \underline{C_r}) = min\{d(mc_i, \underline{C_1}), \ldots, d(mc_i, \underline{C_K})\}$, mc_i is assigned to the boundary region of C_r. The function $d(\cdot)$ denotes the minimum distance between a boundary micro-cluster and the core region of a cluster.

In summary, we use a micro-cluster-based three-way clustering strategy to reconstruct complete three-way clusters step-by-step. The initial core regions of clusters are constructed first. Then, the boundary micro-clusters are assigned in two steps to obtain the final core and boundary regions of clusters. It is demonstrated that the strategy effectively improves the clustering quality of boundary-ambiguous clusters in data streams due to the experimental results in Sec. V.

E. Algorithm and Complexity Analysis

The complete pseudo-code of the proposed TWStream algorithm is shown as Algorithms 1-2.

The storage requirements and time constraints are met in a stream clustering application. The space complexity of TWStream depends on the dimensionality of data objects in $\mathbf{X}$, the number of micro-clusters and the size of the *augmented knn graph*. Assume that $M_t \in \mathbb{R}^d (M_t \ll n)$ is the number of micro-clusters in memory at time t. The memory space to store all micro-clusters is $O(dM_t)$. The *augmented knn graph* is implemented by an adjacency list. In the worst case, the space complexity of the *augmented knn graph* is $O(2kdM_t)$. Therefore, the overall space complexity of TWStream is $O(dM_t) + O(2kdM_t) \approx O(2kdM_t)$.

Algorithm 1 shows that the proposed algorithm consists of two stages. In the online stage, it is necessary to traverse all micro-clusters to find the closest micro-cluster to the new data object, which takes $O(dM_t)$. The worst-case complexity of decaying micro-clusters and detecting the statuses of micro-clusters is $O(M_t)$. Inserting a vertex (based on *Binary Insertion Sort*) into the *augmented knn graph* (Line 29) takes $O(M_t log2k)$. Deleting a vertex from the *augmented knn graph* (Line 16) takes $O(M_t)$. Hence, the time complexity of updating the *augmented knn graph* is $O(M_t log2k)$. Updating $\mathbf{V}$ and the *Outlier Pool* takes $O(1)$. Therefore, the time complexity of the online stage is $O(dM_t) + O(M_t log2k) = O((d + log2k)M_t)$.

In the offline stage, each micro-cluster's boundary confidence must be calculated first (Lines 38-40). The calculation of direction dispersion requires $O(kM_t)$. The time to calculate dimension skewness is $O(kdM_t)$. Calculating the decay-based kernel density takes $O(kM_t)$. Hence, the complexity of calculating the boundary confidence of each

micro-cluster is $O(kdM_t)$. The peeling operation (Line 41) takes $O(M_t)$. As shown in Algorithm 2, the complexity of the mutual reachability-based clustering approach (Lines 1-5) is $O(M_t + 2kM_t)$. The three-way assignment approach (Lines 6-35) takes $O(kM_t) + O(KM_t)$. Hence, the time complexity of the micro-cluster-based three-way clustering strategy is $O((3k + K)M_t)$. Therefore, the time complexity of the offline stage is $O(kdM_t)+O(M_t)+O((3k+K)M_t) \approx O((kd + K)M_t)$.

In conclusion, the overall time complexity of TWStream is about $O((kd + log2k + K)M_t)$.

Algorithm 1: TWStream

Input: $\mathbf{X}$, r, λ, β, k, τ;
Output: $\mathbf{C} = \{[\underline{C_1}, \overline{C_1}], \ldots, [\underline{C_K}, \overline{C_K}]\}$;
1 $t \leftarrow 0$, $\mathbf{V} \leftarrow \emptyset$;
2 **foreach** $x_i \in \mathbf{X}$ **do**
// Online Stage
3 Find the nearest $mc_i \in \mathbf{V}$ to x_i;
4 **if** $d(x_i, mc_i) \leq r$ **then**
5 Update the weight of mc_i by Eq. (4);
6 **if** $d(x_i, mc_i) \leq r/2$ **then**
7 Update the center of mc_i by Eq. (5);
8 Update *augmented knn graph*;
9 **end**
10 **end**
11 **foreach** $mc_i \in \mathbf{V}$ **do**
12 Decay the weight of mc_i;
13 **if** $W(mc_i, t) < W_{min}$ **then**
14 $\mathbf{V} \leftarrow \mathbf{V} \setminus \{mc_i\}$;
15 *Outlier Pool* caches mc_i;
16 Remove mc_i from *augmented knn graph*;
17 **end**
18 **end**
19 **if** $d(x_i, mc_i) > r$ **then**
20 Find the nearest mc_j in *Outlier Pool* to x_i;
21 **if** $d(x_i, mc_j) \leq r$ **then**
22 Update the weight of mc_j by Eq. (4);
23 **if** $d(x_i, mc_j) \leq r/2$ **then**
24 Update the center of mc_j by Eq. (5);
25 **end**
26 **if** $W(mc_j, t) \geq W_{min}$ **then**
27 Remove mc_j from *Outlier Pool*;
28 $\mathbf{V} \leftarrow \mathbf{V} \cup \{mc_j\}$;
29 Insert mc_j into *augmented knn graph*;
30 **end**
31 **end**
32 **end**
33 **if** $d(x_i, mc_i) > r$ *and* $d(x_i, mc_j) > r$ **then**
34 $mc_t \leftarrow$ Create a new micro-cluster by x_i;
35 *Outlier Pool* caches mc_t;
36 **end**
// Offline Stage
37 **if** *a clustering request arrives* **then**
38 **foreach** $mc_p \in \mathbf{V}$ **do**
39 Calculate ϕ_p by Definitions 4-7;
40 **end**
41 Obtain $\mathbf{V}_{border}$, $\mathbf{V}_{core}$ by Eqs. (16)-(17);
42 $\mathbf{C} \leftarrow$ call function **MC-TWC**($\mathbf{V}_{core}$, $\mathbf{V}_{border}$, k);
43 Output $\mathbf{C}$;
44 **end**
45 $t \leftarrow t + 1$;
46 **end**

Algorithm 2: MC-TWC

Input: $\mathbf{V}_{core}$, $\mathbf{V}_{border}$, k;
Output: $\mathbf{C} = \{[\underline{C_1}, \overline{C_1}], \ldots, [\underline{C_K}, \overline{C_K}]\}$;
// Clustering core micro-clusters by mutual reachability
1 **foreach** $mc_i, mc_j \in \mathbf{V}_{core}$ **do** // $i \neq j$
2 **if** $mc_j \in \mathrm{MN}_k(mc_i, t)$ **then**
3 $\underline{C_i} \leftarrow \underline{C_i} \cup \{mc_i, mc_j\}$;
4 **end**
5 **end**
// Initial three-way assignment
6 **foreach** $mc_i \in \mathbf{V}_{border}$ **do**
7 **if** $\mathrm{N}_k(mc_i, t) \cap \mathbf{V}_{core} \neq \emptyset$ **then**
8 **if** $|\{\mathrm{C_s}|\mathrm{P}(\mathrm{C_s}|mc_i) > 0, \mathrm{C_s} \in \mathbf{C}\}| = 1$ **then**
9 **if** $\mathrm{P}(\mathrm{C}_r|mc_i) \geq 1/\mathrm{K}$ **then**
10 $\underline{C_r} \leftarrow \underline{C_r} \cup \{mc_i\}$;
11 **end**
12 **else if** $0 < \mathrm{P}(\mathrm{C}_r|mc_i) < 1/\mathrm{K}$ **then**
13 $BND(C_r) \leftarrow BND(C_r) \cup \{mc_i\}$;
14 **end**
15 **end**
16 **else if** $|\{\mathrm{C_s}|\mathrm{P}(\mathrm{C_s}|mc_i) > 0, \mathrm{C_s} \in \mathbf{C}\}| > 1$ **then**
17 $P(C_r|mc_i) = max(P(\mathbf{C}|mc_i))$;
18 **if** $\exists p$, *s.t.* $\mathrm{P}(\mathrm{C}_r|mc_i) - \mathrm{P}(\mathrm{C}_p|mc_i) < 1/\mathrm{K}$ **then**
19 $BND(C_r) \leftarrow BND(C_r) \cup \{mc_i\}$;
20 **foreach** $q \in \{p|\mathrm{P}(\mathrm{C}_r|mc_i) - \mathrm{P}(\mathrm{C}_p|mc_i) < 1/\mathrm{K}\}$ **do**
21 $BND(C_q) \leftarrow BND(C_q) \cup \{mc_i\}$;
22 **end**
23 **end**
24 **else**
25 $\underline{C_r} \leftarrow \underline{C_r} \cup \{mc_i\}$;
26 **end**
27 **end**
28 **end**
29 **end**
// Further three-way assignment
30 **foreach** $mc_i \in \mathbf{V}_{border}$ **do**
31 **if** $\mathrm{N}_k(mc_i, t) \cap \mathbf{V}_{core} = \emptyset$ **then**
32 $d(mc_i, \underline{C_r}) = min\{d(mc_i, \underline{C_1}), \ldots, d(mc_i, \underline{C_K})\}$;
33 $BND(C_r) \leftarrow BND(C_r) \cup \{mc_i\}$;
34 **end**
35 **end**
36 **return** $\{[\underline{C_1}, \overline{C_1}], \ldots, [\underline{C_K}, \overline{C_K}]\}$.

V. Experimental Results

A. Preparations

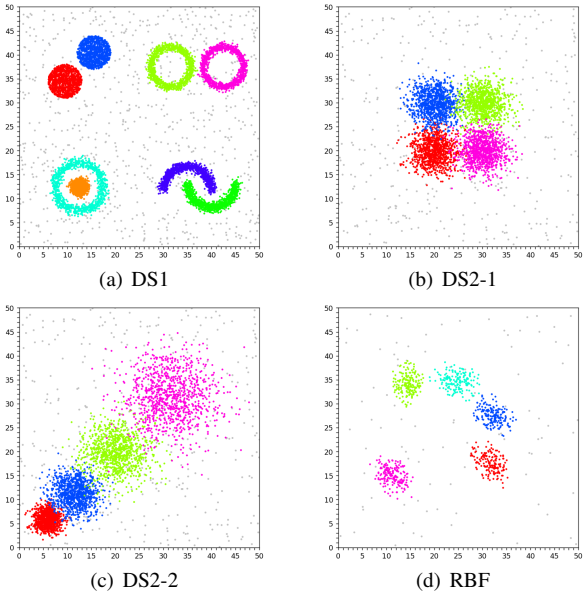
To evaluate the performance of TWStream, we conduct experiments on 15 different datasets. Table I provides the details about 5 synthetic datasets and 10 real-world datasets. All experiments are conducted on a Lenovo Erazer Y40-70 (Intel i5-4210U, 4 cores, 1 thread/core, 2.40GHz and 12GB RAM) with Win10. The TWStream algorithm and nine comparison algorithms are implemented in Java 8.

1) Datasets: Due to page limitations, descriptions of 5 synthetic datasets and 10 real-world datasets are provided in the Supplementary Material. Data distributions of the three 2-dimensional synthetic datasets are illustrated in Fig. 5. In our experiments, we subject the mentioned datasets to a normalization process.

2) Comparison Algorithms: To evaluate the performance of our algorithm, we compare it with nine other density-based clustering algorithms, including micro-cluster-based approaches and grid-based approaches. EDMStream [22], CEDAS [13], DBSTREAM [12], HDDStream [19], and DenStream [17] are micro-cluster-based approaches. Among them, EDMStream and CEDAS are two state-of-the-art online stream clustering algorithms. DBSTREAM is an efficient two-stage clustering algorithm using graph structures. DenStream is a streaming variant of DBSCAN. HDDStream provides a

TABLE I: Datasets used in experiments.

Data Streams		#Instances	#Features	#Clusters
Synthetic	DS1	220,000	2	8
	DS2	220,000	2	8
	RBF	400,000	2	5
	Hyperplane	100,000	10	5
	SynEDC	400,000	40	20
Real-world	NOAAweather	18,159	8	2
	Powersupply	29,928	2	24
	Adult	32,561	6	2
	Electricity	45,312	6	2
	Insects	57,018	33	6
	Rialto	82,250	27	10
	Airlines	539,383	7	2
	Poker	829,201	10	10
	Covertype	581,012	54	7
	Sensor	2,219,803	5	54



(a) DS1 (b) DS2-1 (c) DS2-2 (d) RBF

Fig. 5: Data distributions of synthetic datasets.

suitable clustering solution for high-dimensional data streams. D-Stream [20], MR-Stream [10], MuDi-Stream [11], and ESA-Stream [24] are grid-based approaches. Specifically, D-Stream is a grid-based baseline algorithm, whereas MR-Stream is an optimisation algorithm using multi-resolution grids. MuDi-Stream combines grids and micro-clusters and employs the notion of local cluster density. ESA-Stream is a parameter-adaptive algorithm for processing high-dimensional data streams.

3) Parameter Setup: TWStream has five parameters, namely, the radius of micro-clusters (r), the decay factor (λ), the weight threshold factor (β), the number of nearest neighbors (k), and the boundary percentage factor (τ). As all datasets are normalized in advance, r is set to range from 0.001 to 1. In this study, the value of λ is fixed at 0.0028. As for β, its value ranges from 0 to 1. It is worth noting that a larger β value results in fewer active micro-clusters. As a general rule, we typically set β to $(1-2^{-\lambda}+0.0001)$. The value range of k is set from 4 to 16. The value range of τ is set from 0.5 to 0.8. The setting of the parameters k and τ is described in detail in Sec. V-D. The ranges of parameter values for all algorithms are shown in the Supplementary Material (see Table S1). The clustering results on all datasets are the optimal parameter combinations for the algorithms.

4) Evaluation Metrics: In our experiments, we use three metrics to evaluate the performance of TWStream and comparison algorithms. These metrics include Purity [44], *Normalized Mutual Information* (NMI) [45], and *Adjusted Rand Index* (ARI) [46]. A more detailed explanation of evaluation metrics is provided in the Supplementary Material.

B. Comparison Experiments

This section compares the clustering results of all algorithms on each dataset. In the Supplementary Material, Figs. S3-S4 illustrate the figures of the clustering results of all the algorithms on DS1 and DS2, respectively. A comparison of the performance of each algorithm on each dataset is presented in Table II, where bolded letters indicate the best results for each metric.

1) Comparison Results on Synthetic Datasets: As can be observed in the figures of the clustering results of all the algorithms on DS1 and DS2 (see Figs. S3-S4 in the Supplementary Material), TWStream reconstructs the clusters well. Since the four groups of clusters in the DS1 dataset appear sequentially, we illustrate the clustering results of all algorithms on DS1 at four different times (t1<t2<t3<t4). The first group comprises two circular clusters with uniform distributions located close to one another. TWStream obtains the correct clustering results. D-Stream and DenStream merge the two clusters incorrectly. In the second group, there are two very close rings. The two ring clusters are successfully distinguished by TWStream. CEDAS and D-Stream fail to recognise these two clusters. DBSTREAM, HDDStream and MuDi-Stream over-segment the clusters. In the third group, a ring wraps around a small circular cluster. TWStream identifies the two clusters with a nested relationship successfully. EDMStream, DBSTREAM, HDDStream, and MuDi-Stream all suffer from over-segmentation. D-Stream and MR-Stream clustering fails. ESA-Stream over-segments the ring cluster and incorrectly merges it with the small internal circular cluster. A pair of crescent-shaped clusters make up the last group. Our algorithm exhibits successful results. However, EDMStream, HDDStream, D-Stream, MR-Stream, and MuDi-Stream yield unsatisfactory outcomes. From these results, it is evident that TWStream can filter outliers, recognize arbitrarily shaped clusters, and handle concept drifts effectively.

There are two groups of clusters in DS2 that occur consecutively. The clustering results for all algorithms at times t1 and t2 (t1<t2), respectively. The results demonstrate that TWStream is capable of identifying clusters with ambiguous or even partially overlapping boundaries. According to our analysis, the ability of TWStream to handle clusters with ambiguous boundaries may come from the introduction of boundary confidence and the use of the micro-cluster-based three-way clustering strategy. In addition, the clustering results at time t2 demonstrate the ability of TWStream to recognise

TABLE II: Performance comparison of algorithms on synthetic and real-world datasets.

Dataset	Metric	TWStream	EDMStream	CEDAS	DBSTREAM	HDDStream	DenStream	D-Stream	MR-Stream	MuDi-Stream	ESA-Stream
DS1	Purity	**0.985**	**0.957**	0.910	0.602	0.895	0.778	0.795	0.762	0.708	0.495	0.811	0.851	0.498	0.001	0.830	0.328	0.755	0.522	0.949	0.807
	NMI	**0.969**	3.567	0.548	**2.261**	0.789	3.754	0.740	2.659	0.419	13.256	0.866	4.477	0.001	23.914	0.309	5.223	0.480	6.547	0.834	4.380
	ARI	**0.969**	3.567	0.548	**2.261**	0.789	3.754	0.740	2.659	0.419	13.256	0.866	4.477	0.001	23.914	0.309	5.223	0.480	6.547	0.834	4.380
	Time(s)	**0.969**	3.567	0.548	**2.261**	0.789	3.754	0.740	2.659	0.419	13.256	0.866	4.477	0.001	23.914	0.309	5.223	0.480	6.547	0.834	4.380
DS2	Purity	**0.968**	**0.881**	0.767	0.755	0.498	0.353	0.860	0.583	0.590	0.476	0.699	0.612	0.322	0.087	0.501	0.373	0.695	0.480	0.834	0.784
	NMI	**0.916**	3.554	0.724	**1.471**	0.217	4.719	0.488	3.423	0.210	13.649	0.515	2.719	0.034	11.788	0.282	4.725	0.309	6.837	0.750	4.090
	ARI	**0.916**	3.554	0.724	**1.471**	0.217	4.719	0.488	3.423	0.210	13.649	0.515	2.719	0.034	11.788	0.282	4.725	0.309	6.837	0.750	4.090
	Time(s)	**0.916**	3.554	0.724	**1.471**	0.217	4.719	0.488	3.423	0.210	13.649	0.515	2.719	0.034	11.788	0.282	4.725	0.309	6.837	0.750	4.090
RBF	Purity	**0.827**	**0.812**	0.778	0.757	0.776	0.707	0.769	0.675	0.768	0.589	0.755	0.664	0.539	0.252	0.760	0.757	0.498	0.183	0.658	0.605
	NMI	**0.740**	5.852	0.675	**3.971**	0.637	8.155	0.597	4.645	0.489	30.046	0.587	4.995	0.183	54.615	0.690	8.774	0.085	23.235	0.570	9.340
	ARI	**0.740**	5.852	0.675	**3.971**	0.637	8.155	0.597	4.645	0.489	30.046	0.587	4.995	0.183	54.615	0.690	8.774	0.085	23.235	0.570	9.340
	Time(s)	**0.740**	5.852	0.675	**3.971**	0.637	8.155	0.597	4.645	0.489	30.046	0.587	4.995	0.183	54.615	0.690	8.774	0.085	23.235	0.570	9.340
Hyperplane	Purity	0.721	**0.184**	0.601	0.062	0.546	0.012	0.648	0.136	0.553	0.120	0.542	0.133	0.399	0.107	0.643	0.101	0.498	0.107	**0.738**	0.091
	NMI	**0.120**	**2.784**	0.026	2.891	0.004	5.332	0.093	3.210	0.039	13.234	0.047	6.748	0.089	14.992	0.022	6.363	0.035	7.796	0.074	7.800
	ARI	**0.120**	**2.784**	0.026	2.891	0.004	5.332	0.093	3.210	0.039	13.234	0.047	6.748	0.089	14.992	0.022	6.363	0.035	7.796	0.074	7.800
	Time(s)	**0.120**	**2.784**	0.026	2.891	0.004	5.332	0.093	3.210	0.039	13.234	0.047	6.748	0.089	14.992	0.022	6.363	0.035	7.796	0.074	7.800
SynEDC	Purity	0.928	0.801	0.923	0.768	0.516	0.315	**0.930**	**0.852**	0.528	0.675	0.344	0.549	0.364	0.573	0.863	0.728	0.795	0.409	0.881	0.757
	NMI	**0.789**	5.225	0.668	7.196	0.209	26.730	0.639	14.889	0.412	267.356	0.308	**5.113**	0.312	493.727	0.628	14.196	0.255	365.343	0.635	36.300
	ARI	**0.789**	5.225	0.668	7.196	0.209	26.730	0.639	14.889	0.412	267.356	0.308	**5.113**	0.312	493.727	0.628	14.196	0.255	365.343	0.635	36.300
	Time(s)	**0.789**	5.225	0.668	7.196	0.209	26.730	0.639	14.889	0.412	267.356	0.308	**5.113**	0.312	493.727	0.628	14.196	0.255	365.343	0.635	36.300
NOAAweather	Purity	**0.769**	0.197	0.747	0.066	0.684	0.007	0.721	0.066	0.668	0.066	0.642	0.056	0.582	0.071	0.701	0.085	0.663	0.052	0.725	**0.205**
	NMI	0.190	**0.421**	0.003	2.317	0.003	4.323	0.049	2.980	0.058	12.550	0.046	4.342	0.036	45.324	0.027	4.432	0.004	6.543	**0.205**	7.493
	ARI	0.190	**0.421**	0.003	2.317	0.003	4.323	0.049	2.980	0.058	12.550	0.046	4.342	0.036	45.324	0.027	4.432	0.004	6.543	**0.205**	7.493
	Time(s)	0.190	**0.421**	0.003	2.317	0.003	4.323	0.049	2.980	0.058	12.550	0.046	4.342	0.036	45.324	0.027	4.432	0.004	6.543	**0.205**	7.493
Powersupply	Purity	0.407	**0.459**	0.411	0.171	0.369	0.316	0.402	0.314	0.367	0.263	0.382	0.282	0.299	0.105	**0.421**	0.120	0.377	0.209	0.384	0.208
	NMI	**0.120**	1.291	0.043	**1.129**	0.080	3.863	0.080	1.581	0.088	6.878	0.074	2.946	0.015	6.229	0.017	3.986	0.065	4.382	0.052	3.664
	ARI	**0.120**	1.291	0.043	**1.129**	0.080	3.863	0.080	1.581	0.088	6.878	0.074	2.946	0.015	6.229	0.017	3.986	0.065	4.382	0.052	3.664
	Time(s)	**0.120**	1.291	0.043	**1.129**	0.080	3.863	0.080	1.581	0.088	6.878	0.074	2.946	0.015	6.229	0.017	3.986	0.065	4.382	0.052	3.664
Adult	Purity	**0.890**	0.120	0.878	0.030	0.865	0.028	0.887	0.016	0.826	0.047	0.821	0.048	0.841	0.043	0.874	0.148	0.801	0.037	0.806	**0.177**
	NMI	**0.165**	**1.203**	0.081	1.590	0.034	5.688	0.035	1.662	0.041	10.498	0.035	4.092	0.053	49.452	0.105	6.559	0.026	6.968	0.114	7.750
	ARI	**0.165**	**1.203**	0.081	1.590	0.034	5.688	0.035	1.662	0.041	10.498	0.035	4.092	0.053	49.452	0.105	6.559	0.026	6.968	0.114	7.750
	Time(s)	**0.165**	**1.203**	0.081	1.590	0.034	5.688	0.035	1.662	0.041	10.498	0.035	4.092	0.053	49.452	0.105	6.559	0.026	6.968	0.114	7.750
Electricity	Purity	0.779	0.141	0.737	0.114	0.767	0.112	0.725	0.114	0.731	0.105	0.697	0.082	0.526	0.020	**0.784**	**0.173**	0.707	0.059	0.692	0.692
	NMI	**0.118**	**1.750**	0.082	1.672	0.091	5.062	0.074	2.958	0.099	28.730	0.058	3.497	0.007	6.593	0.099	5.735	0.018	20.654	0.049	5.590
	ARI	**0.118**	**1.750**	0.082	1.672	0.091	5.062	0.074	2.958	0.099	28.730	0.058	3.497	0.007	6.593	0.099	5.735	0.018	20.654	0.049	5.590
	Time(s)	**0.118**	**1.750**	0.082	1.672	0.091	5.062	0.074	2.958	0.099	28.730	0.058	3.497	0.007	6.593	0.099	5.735	0.018	20.654	0.049	5.590
Insects	Purity	**0.680**	0.364	0.597	0.336	0.528	0.026	0.633	0.231	0.535	0.056	0.586	0.059	0.409	0.117	0.581	**0.374**	0.407	0.134	0.561	0.050
	NMI	**0.221**	**2.578**	0.119	2.634	0.005	5.270	0.109	2.984	0.010	15.297	0.012	2.608	0.006	118.596	0.145	4.875	0.021	12.190	0.028	10.744
	ARI	**0.221**	**2.578**	0.119	2.634	0.005	5.270	0.109	2.984	0.010	15.297	0.012	2.608	0.006	118.596	0.145	4.875	0.021	12.190	0.028	10.744
	Time(s)	**0.221**	**2.578**	0.119	2.634	0.005	5.270	0.109	2.984	0.010	15.297	0.012	2.608	0.006	118.596	0.145	4.875	0.021	12.190	0.028	10.744
Rialto	Purity	**0.447**	0.374	0.349	0.255	0.408	0.193	0.303	0.231	0.399	0.227	0.375	0.188	0.350	0.188	0.401	**0.407**	0.367	0.195	0.401	0.264
	NMI	**0.296**	**3.634**	0.152	3.719	0.121	11.391	0.125	5.922	0.065	13.876	0.045	10.980	0.022	47.329	0.235	5.077	0.036	9.233	0.218	8.459
	ARI	**0.296**	**3.634**	0.152	3.719	0.121	11.391	0.125	5.922	0.065	13.876	0.045	10.980	0.022	47.329	0.235	5.077	0.036	9.233	0.218	8.459
	Time(s)	**0.296**	**3.634**	0.152	3.719	0.121	11.391	0.125	5.922	0.065	13.876	0.045	10.980	0.022	47.329	0.235	5.077	0.036	9.233	0.218	8.459
Airlines	Purity	**0.720**	**0.200**	0.626	0.111	0.642	0.062	0.654	0.059	0.601	0.141	0.641	0.133	0.627	0.014	0.644	0.088	0.598	0.033	0.687	0.078
	NMI	**0.179**	8.378	0.007	**6.973**	0.009	22.858	0.014	7.218	0.045	28.673	0.069	13.967	0.001	294.151	0.073	11.260	0.003	241.618	0.046	18.790
	ARI	**0.179**	8.378	0.007	**6.973**	0.009	22.858	0.014	7.218	0.045	28.673	0.069	13.967	0.001	294.151	0.073	11.260	0.003	241.618	0.046	18.790
	Time(s)	**0.179**	8.378	0.007	**6.973**	0.009	22.858	0.014	7.218	0.045	28.673	0.069	13.967	0.001	294.151	0.073	11.260	0.003	241.618	0.046	18.790
Poker	Purity	**0.750**	**0.250**	0.654	0.061	0.675	0.089	0.694	0.105	0.657	0.085	0.661	0.095	0.636	0.062	0.601	0.079	0.599	0.096	0.683	0.083
	NMI	**0.250**	**26.941**	0.024	28.92	0.033	40.872	0.005	30.350	0.005	45.575	0.006	33.736	0.005	349.800	0.004	40.497	0.027	640.565	0.018	47.800
	ARI	**0.250**	**26.941**	0.024	28.92	0.033	40.872	0.005	30.350	0.005	45.575	0.006	33.736	0.005	349.800	0.004	40.497	0.027	640.565	0.018	47.800
	Time(s)	**0.250**	**26.941**	0.024	28.92	0.033	40.872	0.005	30.350	0.005	45.575	0.006	33.736	0.005	349.800	0.004	40.497	0.027	640.565	0.018	47.800
Covertype	Purity	0.887	**0.315**	0.875	0.308	0.894	0.297	**0.902**	0.302	0.852	0.268	0.849	0.260	0.781	0.208	0.891	0.307	0.781	0.200	0.885	0.295
	NMI	**0.122**	14.640	0.111	**14.563**	0.097	31.908	0.102	24.879	0.097	34.754	0.077	19.297	0.064	242.359	**0.122**	19.351	0.054	63.257	0.115	38.535
	ARI	**0.122**	14.640	0.111	**14.563**	0.097	31.908	0.102	24.879	0.097	34.754	0.077	19.297	0.064	242.359	**0.122**	19.351	0.054	63.257	0.115	38.535
	Time(s)	**0.122**	14.640	0.111	**14.563**	0.097	31.908	0.102	24.879	0.097	34.754	0.077	19.297	0.064	242.359	**0.122**	19.351	0.054	63.257	0.115	38.535
Sensor	Purity	**0.761**	**0.847**	0.447	0.582	0.511	0.640	0.388	0.545	0.520	0.773	0.494	0.810	0.261	0.354	0.515	0.678	0.501	0.266	0.450	0.419
	NMI	**0.642**	25.965	0.284	30.563	0.331	36.975	0.274	32.455	0.592	45.101	0.561	39.423	0.076	3561.896	0.334	**19.911**	0.046	73.788	0.193	39.765
	ARI	**0.642**	25.965	0.284	30.563	0.331	36.975	0.274	32.455	0.592	45.101	0.561	39.423	0.076	3561.896	0.334	**19.911**	0.046	73.788	0.193	39.765
	Time(s)	**0.642**	25.965	0.284	30.563	0.331	36.975	0.274	32.455	0.592	45.101	0.561	39.423	0.076	3561.896	0.334	**19.911**	0.046	73.788	0.193	39.765

clusters with varying densities. The other algorithms produce poor clustering results on DS2. We conjecture that TWStream's ability to identify multi-density data effectively stems from the *Decay-based Kernel Density* proposed in Definition 6.

Specifically, the following conclusions can be drawn by comparing the first five rows of clustering results in Table II. For the ARI metric, TWStream leads the comparison algorithms on all synthetic datasets. Additionally, TWStream outperforms its competitors on four synthetic datasets (i.e., DS1, DS2, RBF, and Hyperplane) in terms of the Purity and NMI metrics.

2) Comparison Results on Real-world Datasets: Next, we compare the clustering performance of all algorithms on real-world datasets based on the last ten rows in Table II. TWStream has higher ARI values than all other algorithms on nine real-world datasets. In addition, the proposed algorithm achieves the highest performance for Purity and NMI on at least five real-world datasets. Specifically, on the NOAAweather dataset, TWStream achieves slightly lower NMI and ARI scores than ESA-Stream but significantly higher than the third-best algorithm. On the Powersupply dataset, TWStream significantly outperforms the other algorithms in terms of NMI and ARI, although its Purity value is slightly lower than that of MR-Stream. Experimental results on Adult show that our algorithm outperforms others in terms of Purity and ARI metrics. While TWStream has the second-highest NMI score, it is only 0.057 lower than the first place and much higher than the third place. On the Electricity dataset, the proposed algorithm achieves higher scores than other algorithms in terms of ARI scores while falling marginally behind MR-Stream with regard to Purity and NMI scores. On the Insects dataset and Rialto dataset, the proposed algorithm outperforms other algorithms in both Purity and ARI metrics. NMI values are just 0.01 and 0.033 lower than MR-Stream on the above datasets. The proposed algorithm outperforms other competing algorithms on the Airlines dataset. According to experimental results on the Poker dataset, the proposed algorithm improves performance by more than 8% on all evaluation metrics compared to the second-best algorithm. On the Covertype dataset, our algorithm achieves first place in the NMI and ARI metrics. It is slightly below the DBSTREAM algorithm in the Purity metric. TWStream outperforms all competitors on the Sensor dataset.

In summary, the ability of TWStream to reconstruct high-quality clusters in a streaming environment can be attributed to two factors: 1) the detection of boundary confidence and 2) the micro-cluster-based three-way cluetring strategy. The boundary confidence detect cluster boundaries efficiently and reveal potential cores of clusters. It integrates the skewness and sparsity of the data distribution, as well as the evolving trend of the stream. The micro-cluster-based three-way clustering strategy reconstructs potential clusters effectively, which improves the clustering quality of boundary-ambiguous clusters in a stream. In contrast, other stream clustering algorithms do not fully consider the neighbourhood information and cluster structure of the data at a finer granularity. This may reduce the algorithms' ability to handle multi-density clusters with ambiguous boundaries.

3) Running Time Comparison: On real-world datasets, we compare the proposed algorithm's running time with competitors using micro-clusters as summary structures (i.e.,

EDMStream, CEDAS, DBSTREAM, HDDStream, and DenStream). To estimate the running time, we use the average of 20 repeated experiments. Fig. 6 shows that HDDStream is significantly slower than other algorithms. Moreover, Fig. 6 illustrates that TWStream, EDMStream, and DBSTREAM are the three fastest-running algorithms. It is noteworthy that TWStream has achieved first place six times, which is more than EDMStream and DBSTREAM. That means TWStream is usually faster than EDMStream and DBSTREAM. Table II shows the detailed running times of all algorithms on each dataset. It is evident from the analysis above that TWStream has the ability to process a massive amount of data in the stream efficiently.

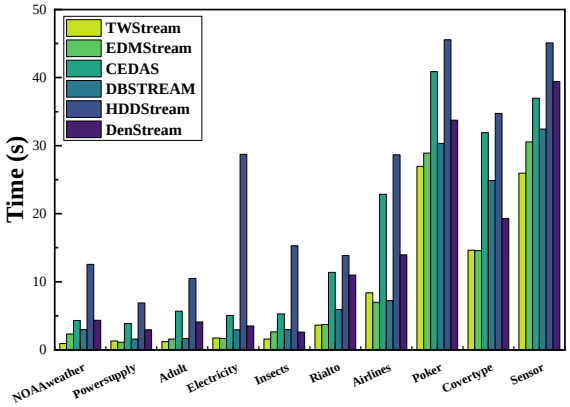


Fig. 6: Running time comparison of algorithms on real-world datasets.

C. Ablation Experiments

In this subsection, we perform ablation experiments to evaluate the impacts of the boundary confidence and the micro-cluster-based three-way clustering strategy, the two key components of our algorithm, on the clustering performance. Firstly, we remove each of the three components of the boundary confidence: direction dispersion (η), dimension skewness (γ), and decay-based kernel density (ρ). This produces modified algorithms named TWStream-η, TWStream-γ, and TWStream-ρ. Then, we remove the boundary confidence (ϕ), substituting it with the inverse of micro-cluster weights, in the proposed algorithm. The modified algorithm is called TWStream-ϕ. Finally, we remove the micro-cluster-based three-way clustering strategy from the proposed algorithm, replacing it with a simple merging approach using adjacent micro-clusters based on a fixed distance threshold. The modified algorithm is called TWStream-3w. The modified algorithms are tested on all of the datasets listed in Table I. The modified algorithms' parameter settings align with those specified in TWStream for each respective dataset.

The detailed results of the ablation experiments are presented in Table III, with the best metric values bolded for each dataset. Furthermore, a better visual representation of the results is provided in Fig. S5 of the Supplementary Material. Observably, all the modified algorithms exhibit lower performance compared to TWStream (i.e., the proposed algorithm). Specifically, TWStream-η shows the slightest degradation in clustering performance, followed by TWStream-γ and TWStream-ρ. TWStream-ϕ presents the significant performance decrements across all datasets when compared to TWStream. This degradation stems from the modified algorithm's neglect of critical factors, including direction dispersion, dimension skewness, and decay-based kernel density. As a result, it encounters difficulties in capturing adequate information required for the accurate identification of micro-clusters. The boundary confidence affects clustering results to a significant extent, which is generally consistent with experimental expectations. These observations show that boundary confidence is not a simple density measure. Instead, it is a complex and comprehensive approach to capturing data distribution. TWStream-3w exhibits the worst performance across all datasets. This demonstrates that the proposed three-way clustering strategy significantly improves clustering performance, particularly when confronted with the challenge of identifying clusters with ambiguous boundaries.

In summary, the ablation experiments validate that the boundary confidence and the micro-cluster-based three-way clustering strategy can enhance the overall clustering performance significantly.

D. Parameter Analysis

In this subsection, we investigate the clustering performance impact of all five parameters of the proposed algorithm: the number of nearest neighbors k, the boundary percentage factor τ, the micro-cluster radius r, the decay factor λ, and the weight threshold factor β. The experiments are conducted on three synthetic datasets, namely DS2, RBF, and Bench[1]. When analyzing one parameter, the other parameters are fixed.

Due to page limitations, we provide details of the parameter analysis process in the Supplementary Material.

VI. Conclusion

In this paper, we develop a data stream clustering algorithm based on the three-way decision theory called TWStream. TWStream implements a two-stage model for density-based clustering. In the online stage, an augmented $k$nn graph is maintained incrementally to accelerate the update of the $k$nn graph. In the offline stage, The boundary confidence is introduced in TWStream to detect cluster boundaries efficiently and reveal potential cluster cores. It takes into account data skewness and sparsity, as well as the evolving trend of the stream. In addition, a micro-cluster-based three-way clustering strategy is applied to reconstruct latent clusters. It improves the clustering quality of boundary-ambiguous clusters in a stream using a mutual reachability-based clustering approach and a three-way assignment approach. Extensive experiments have been conducted to demonstrate the effectiveness and efficiency of the proposed algorithm.

As with most stream clustering approaches based on summary structures, TWStream may suffer from low clustering quality when the dimensionality of the stream is high (see the analysis report in the Supplementary Material). In future work,

[1]https://github.com/Xicks/DataStream_DataSets

TABLE III: Results of ablation experiments.

Algorithm	DS1			DS2			RBF			Hyperplane			SynEDC		
	Purity	NMI	ARI	Purity	NMI	ARI	Purity	NMI	ARI	Purity	NMI	ARI	Purity	NMI	ARI
TWStream	**0.985**	**0.957**	**0.969**	**0.968**	**0.881**	**0.916**	**0.827**	**0.812**	**0.740**	**0.721**	**0.184**	**0.120**	**0.928**	**0.801**	**0.789**
TWStream-η	0.983	0.952	**0.969**	0.959	0.865	0.897	0.819	0.809	0.735	0.521	0.074	0.040	0.925	0.797	0.785
TWStream-γ	0.978	0.945	0.963	0.952	0.857	0.880	0.811	0.805	0.731	0.514	0.077	0.029	**0.928**	0.794	0.782
TWStream-ρ	0.974	0.937	0.958	0.950	0.857	0.882	0.812	0.806	0.734	0.529	0.099	0.036	0.920	0.791	0.779
TWStream-ϕ	0.966	0.919	0.931	0.927	0.831	0.853	0.802	0.783	0.725	0.507	0.071	0.013	0.916	0.788	0.773
TWStream-3w	0.906	0.891	0.783	0.839	0.672	0.616	0.799	0.745	0.678	0.484	0.008	0.007	0.669	0.528	0.512
Algorithm	NOAAweather			Powersupply			Adult			Electricity			Insects		
	Purity	NMI	ARI	Purity	NMI	ARI	Purity	NMI	ARI	Purity	NMI	ARI	Purity	NMI	ARI
TWStream	**0.769**	**0.197**	**0.190**	**0.407**	**0.459**	**0.120**	**0.890**	**0.120**	**0.165**	**0.779**	**0.141**	**0.118**	**0.680**	**0.364**	**0.221**
TWStream-η	**0.769**	**0.197**	**0.190**	0.301	0.445	0.100	0.852	0.077	0.110	0.775	0.141	0.117	0.567	0.363	**0.221**
TWStream-γ	0.753	0.182	0.186	0.297	0.439	0.098	0.856	0.063	0.098	0.774	0.135	0.117	0.569	0.361	0.215
TWStream-ρ	0.749	0.166	0.175	0.292	0.430	0.093	0.852	0.064	0.099	0.768	0.129	0.107	0.562	0.358	0.211
TWStream-ϕ	0.732	0.103	0.069	0.277	0.409	0.082	0.848	0.055	0.083	0.763	0.129	0.103	0.547	0.344	0.195
TWStream-3w	0.661	0.062	0.050	0.205	0.342	0.046	0.826	0.036	0.058	0.724	0.113	0.089	0.528	0.329	0.168
Algorithm	Rialto			Airlines			Poker			Covertype			Sensor		
	Purity	NMI	ARI	Purity	NMI	ARI	Purity	NMI	ARI	Purity	NMI	ARI	Purity	NMI	ARI
TWStream	**0.447**	**0.374**	**0.296**	**0.720**	**0.200**	**0.179**	**0.750**	**0.250**	**0.250**	**0.887**	**0.315**	**0.122**	**0.761**	**0.847**	**0.642**
TWStream-η	0.442	0.371	0.292	0.715	0.191	0.173	0.667	0.244	0.248	0.873	0.299	**0.122**	0.667	0.808	0.545
TWStream-γ	0.443	0.369	0.288	0.716	0.193	0.173	0.663	0.240	0.243	0.886	0.306	0.112	0.733	0.824	0.578
TWStream-ρ	0.439	0.361	0.283	0.708	0.186	0.166	0.664	0.242	0.235	0.864	0.299	0.113	0.716	0.820	0.579
TWStream-ϕ	0.431	0.352	0.270	0.694	0.180	0.161	0.653	0.235	0.212	0.852	0.284	0.108	0.645	0.801	0.528
TWStream-3w	0.396	0.310	0.205	0.633	0.133	0.102	0.646	0.218	0.180	0.837	0.254	0.086	0.550	0.682	0.499

we will strive to improve this weakness. Furthermore, a worthy direction is to research how to perform adaptive tuning of k and τ to adapt to the dynamic evolution of data distribution in a stream. One possible idea is to monitor a performance metric during the execution of the algorithm and adjust the values of k and τ automatically when performance drops are perceived. Another interesting research direction is to extend TWStream to parallel streaming environments. In this way, the algorithm can be applied to a wide range of application scenarios.

REFERENCES

[1] N. Begum and E. Keogh, "Rare time series motif discovery from unbounded streams," *Proceedings of the VLDB Endowment*, vol. 8, no. 2, pp. 149–160, 2014.

[2] L. Cao, Q. Wang, and E. A. Rundensteiner, "Interactive outlier exploration in big data streams," *Proceedings of the VLDB Endowment*, vol. 7, no. 13, pp. 1621–1624, 2014.

[3] H. Huang and S. P. Kasiviswanathan, "Streaming anomaly detection using randomized matrix sketching," *Proceedings of the VLDB Endowment*, vol. 9, no. 3, pp. 192–203, 2015.

[4] Y. Yao, "The Dao of three-way decision and three-world thinking," *International Journal of Approximate Reasoning*, p. 109032, 2023.

[5] Y. Yao and J. Yang, "Granular fuzzy sets and three-way approximations of fuzzy sets," *International Journal of Approximate Reasoning*, vol. 161, p. 109003, 2023.

[6] X. Zhang, Z. Yuan, and D. Miao, "Outlier detection using three-way neighborhood characteristic regions and corresponding fusion measurement," *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[7] A. Shah, N. Azam, E. Alanazi, and J. Yao, "Image blurring and sharpening inspired three-way clustering approach," *Applied Intelligence*, vol. 52, no. 15, pp. 18 131–18 155, 2022.

[8] H. Yu, X. Wang, G. Wang, and X. Zeng, "An active three-way clustering method via low-rank matrices for multi-view data," *Information Sciences*, vol. 507, pp. 823–839, 2020.

[9] M. Du, J. Zhao, J. Sun, and Y. Dong, "M3W: Multistep three-way clustering," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[10] L. Wan, W. K. Ng, X. H. Dang, P. S. Yu, and K. Zhang, "Density-based clustering of data streams at multiple resolutions," *ACM Transactions on Knowledge Discovery from Data*, vol. 3, no. 3, pp. 1–28, 2009.

[11] A. Amini, H. Saboohi, T. Herawan, and T. Y. Wah, "MuDi-Stream: A multi density clustering algorithm for evolving data stream," *Journal of Network and Computer Applications*, vol. 59, pp. 370–385, 2016.

[12] M. Hahsler and M. Bolaños, "Clustering data streams based on shared density between micro-clusters," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 6, pp. 1449–1461, 2016.

[13] R. Hyde, P. Angelov, and A. R. MacKenzie, "Fully online clustering of evolving data streams into arbitrarily shaped clusters," *Information Sciences*, vol. 382, pp. 96–114, 2017.

[14] H. Yu and Y. Wang, "Three-way decisions method for overlapping clustering," in *Proceedings of the 8th International Conference on Rough Sets and Current Trends in Computing, Chengdu, China, August 17-20, 2012*. Springer, 2012, pp. 277–286.

[15] C. C. Aggarwal, S. Y. Philip, J. Han, and J. Wang, "A framework for clustering evolving data streams," in *Proceedings of the 29th International Conference on Very Large Data Bases*, Berlin, Germany, 2003, pp. 81–92.

[16] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," *ACM sigmod record*, vol. 25, no. 2, pp. 103–114, 1996.

[17] F. Cao, M. Estert, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in *Proceedings of SIAM International Conference on Data Mining*, Bethesda, MD, USA, 2006, pp. 328–339.

[18] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Portland, Oregon, USA, 1996, pp. 226–231.

[19] I. Ntoutsi, A. Zimek, T. Palpanas, P. Kröger, and H.-P. Kriegel, "Density-based projected clustering over high dimensional data streams," in *Proceedings of SIAM International Conference on Data Mining*, Anaheim, California, USA, 2012, pp. 987–998.

[20] Y. Chen and L. Tu, "Density-based clustering for real-time stream data," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Jose, California, USA, 2007, pp. 133–142.

[21] L. Tu and Y. Chen, "Stream data clustering based on grid density and attraction," *ACM Transactions on Knowledge Discovery from Data*, vol. 3, no. 3, pp. 1–27, 2009.

[22] S. Gong, Y. Zhang, and G. Yu, "Clustering stream data by exploring the evolution of density mountain," *Proceedings of the VLDB Endowment*, vol. 11, no. 4, pp. 393–405, 2017.

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

[24] Y. Li, H. Li, Z. Wang, B. Liu, J. Cui, and H. Fei, "ESA-Stream: Efficient self-adaptive online data stream clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 2, pp. 617–630, 2022.

[25] H. Yu, L. Chen, and J. Yao, "A three-way density peak clustering method based on evidence theory," *Knowledge-Based Systems*, vol. 211, p. 106532, 2021.

[26] P. Wang and Y. Yao, "CE3: A three-way clustering method based on

mathematical morphology," *Knowledge-Based Systems*, vol. 155, pp. 54–65, 2018.
[27] P. Wang, T. Wu, and Y. Yao, "A three-way adaptive density peak clustering (3W-ADPC) method," *Applied Intelligence*, pp. 1–17, 2023.
[28] C. Sun, M. Du, J. Sun, K. Li, and Y. Dong, "A three-way clustering method based on improved density peaks algorithm and boundary detection graph," *International Journal of Approximate Reasoning*, vol. 153, pp. 239–257, 2023.
[29] H. Yu, L. Chen, J. Yao, and X. Wang, "A three-way clustering method based on an improved dbscan algorithm," *Physica A: Statistical Mechanics and its Applications*, vol. 535, p. 122289, 2019.
[30] Y. Yao and X. Deng, "Sequential three-way decisions with probabilistic rough sets," in *Proceedings of the 10th IEEE International Conference on Cognitive Informatics and Cognitive Computing*, 2011, pp. 120–125.
[31] C. Jiang, Z. Li, and J. Yao, "A shadowed set-based three-way clustering ensemble approach," *International Journal of Machine Learning and Cybernetics*, vol. 13, no. 9, pp. 2545–2558, 2022.
[32] J. Fan, P. Wang, C. Jiang, X. Yang, and J. Song, "Ensemble learning using three-way density-sensitive spectral clustering," *International Journal of Approximate Reasoning*, vol. 149, pp. 70–84, 2022.
[33] G. Krempl, I. Žliobaite, D. Brzeziński, E. Hüllermeier, M. Last, V. Lemaire, T. Noack, A. Shaker, S. Sievi, M. Spiliopoulou *et al.*, "Open challenges for data stream mining research," *ACM SIGKDD Explorations Newsletter*, vol. 16, no. 1, pp. 1–10, 2014.
[34] D. Barbará, "Requirements for clustering data streams," *ACM SIGKDD Explorations Newsletter*, vol. 3, no. 2, pp. 23–27, 2002.
[35] M. Khalilian and N. Mustapha, "Data stream clustering: Challenges and issues," *arXiv preprint arXiv:1006.5261*, 2010.
[36] Z. Kang, C. Peng, Q. Cheng, X. Liu, X. Peng, Z. Xu, and L. Tian, "Structured graph learning for clustering and semi-supervised classification," *Pattern Recognition*, vol. 110, p. 107627, 2021.
[37] H. Yu, C. Zhang, and G. Wang, "A tree-based incremental overlapping clustering method using the three-way decision theory," *Knowledge-Based Systems*, vol. 91, pp. 189–203, 2016.
[38] H. Averbuch-Elor, N. Bar, and D. Cohen-Or, "Border-peeling clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 7, pp. 1791–1797, 2020.
[39] D. Peng, Z. Gui, D. Wang, Y. Ma, Z. Huang, Y. Zhou, and H. Wu, "Clustering by measuring local direction centrality for data with heterogeneous density and weak connectivity," *Nature Communications*, vol. 13, p. 5455, 2022.
[40] Z. Kang, Z. Lin, X. Zhu, and W. Xu, "Structured graph learning for scalable subspace clustering: From single view to multiview," *IEEE Transactions on Cybernetics*, vol. 52, no. 9, pp. 8976–8986, 2022.
[41] L. Breiman, W. Meisel, and E. Purcell, "Variable kernel estimates of multivariate densities," *Technometrics*, vol. 19, no. 2, pp. 135–144, 1977.
[42] Z. Lin, Z. Kang, L. Zhang, and L. Tian, "Multi-view attributed graph clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 2, pp. 1872–1880, 2023.
[43] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2013, pp. 160–172.
[44] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," 2000.
[45] M. Meilă, "Comparing clusterings—an information based distance," *Journal of Multivariate Analysis*, vol. 98, no. 5, pp. 873–895, 2007.
[46] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, pp. 193–218, 1985.

Jiarui Sun received B.S. degree from Jiangsu Normal University, Xuzhou, China, in 2019. He is currently pursuing the master degree with the School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, China.

His research interests include big data analysis and data stream clustering.

Mingjing Du (Member, IEEE) received the Ph.D. degree in computer science from the China University of Mining and Technology, Xuzhou, China, in 2018. He is currently an associate professor with the School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, China.

His research interests include cluster analysis and three-way decisions. For more information, see https://dumingjing.github.io/

Zhenkang Lew is an undergraduate student with the School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, China.

His research interest includes machine learning and cluster analysis.

Yongquan Dong received the Ph.D. degree in computer science from Shandong University, Jinan, China, in 2010. He is currently a professor with the School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, China.

His research interests include web information integration and web data management.