A Review on Data Stream Clustering Algorithms over Sliding Windows

K. Shyam Sunder Reddy
Department of IT, Vasavi College of Engineering, Hyderabad

Abstract: Density-based clustering technique is a well-known and prominent method for finding hidden patterns from the data streams. It can find clusters with arbitrary shape, detect noisy/outlier data, and does not need the number of clusters in advance. While traditional clustering techniques based on density considered only static data, recent research issues and real-world domains in mining data streams should handle continuous, unbounded data streams, arriving at rapid speed. Data streams are evolving over time and the amount of data is unlimited. The main problem related with streaming data is their storage, management, analysis, and retrieval. This paper presents a survey of different clustering techniques for streaming data over sliding windows, providing an extensive comparison among the various methods.
Keywords: **Data stream, Density-based clustering, Micro-clustering, Sliding windows.**

1. INTRODUCTION

In today's world, data is an important term and it is being generated at a massive rate. Over 90% of the current data in the world has been created in the last two years only. The world gradually connects with emerging technologies, and growing electronic devices, that will enhance massive data creation in the subsequent years. In this big data era, the data is generated continuously and often at extremely huge volumes and velocities. This kind of data generated at a high speed is called a *data stream*.

Definition: Data Stream is an ordered series of d-dimensional data point's P_1... P_n arriving at time stamps T_1... T_n, that can be read once using limited processing capabilities such as limited time and limited memory.

Some of the examples of real-time data stream applications consists of smart cities, weather monitoring, energy utilization monitoring, financial transactions, monitoring ogf urban traffic, industrial sensor data monitoring, healthcare systems, earthquake monitoring, and stock trading live updates [1]. Usually these applications are handling with a huge amount of data over a period of time. In many cases, the data points are read only once due to the enormous availability and velocity of data, and is not possible to store the entire data in main memory for further analysis. The huge amount of data created by real-world applications need to be analysed systematically and processed within limited time and memory to find the hidden patterns in the data. Data stream clustering has attracted many researchers recently, and one of the main issues that have been considered in this domain is finding patterns/clusters in the streaming data.

Clustering is an unsupervised learning task which deals with huge amount data that do not have any labels attached to them [2]. It is used in many real-world domains to retrieve the useful information from the huge amount of data. Clustering's main goal is to discover the group of cohesive objects in the given input domain. It is the task of finding natural and useful patterns of the data objects such that the inter-cluster similarity is minimized and the intra-cluster similarity is maximized. Moreover, all the data objects need not fall into clusters, and few outlier data points do not fall into any of the clusters.

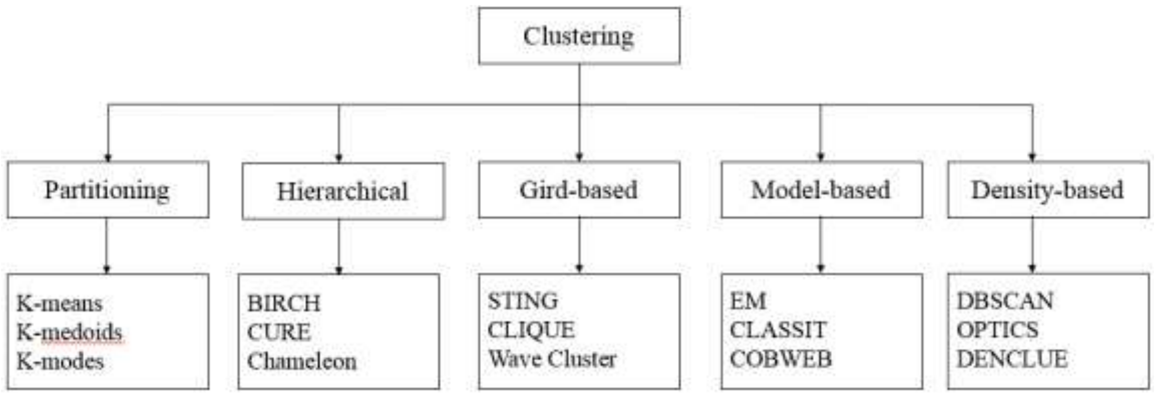

Figure 1: A Taxonomy of Clustering Algorithms

1.1. Taxonomy of Clustering Algorithms:

Clustering techniques can be grouped into several categories [3] based on various criteria: Partitioning clustering, Grid-based clustering, Hierarchical clustering, Density-based clustering, and Model-based clustering. Figure 1 shows the classification of different clustering techniques. Table 1 shows the advantages and disadvantages of various clustering techniques. Among all the clustering methods, the density-based clustering methods are the most appropriate and prominent methods for clustering data streams because of the following main features. A density-based clustering technique:

- can discover arbitrary shape clusters that are noiseless
- can handle outlier data points
- can attain clusters in single scan
- has no assumption on the number of clusters

Hence, the density-based method became a prominent class for stream data clustering.

Table 1: Advantages and Disadvantages of Clustering Methods

Method	Advantages	Disadvantages
Partitioning	- Very simple to implement - Computationally fast clustering method. - Applicable for spherical shape clusters that are well-separated.	- Requires the number of clusters in advance - Cannot find arbitrary shape clusters - Not able to handle outlier data points.
Gird-Based	- Fast processing time. - Significant reduction in time complexity, especially for very large data sets. - Can handle outlier data objects	- Not suitable for High-dimensional data sets. - Requires several parameters in advance.
Hierarchical	- Does not need the number of clusters in advance - Clustering accuracy is high - Can attain arbitrary shape clusters.	- Once a merge or split step is completed, it cannot be undone. - Expensive for high-dimensional data sets.
Density-Based	- Can find arbitrary shape clusters - Can handle outliers - Does not need the number of clusters in advance	- Requires various parameters in advance. - Does not work well in multi-density data.
Model-Based	- Easy to implement and robust to noisy data. - Specifies the number of clusters automatically based on standard statistics	- High complexity - Not applicable for clustering large data sets.

2. DATA STREAM CLUSTERING

The data stream clustering domain is getting a lot of significance and is continuing at a high pace with new techniques, procedures, and findings in various fields such as weather monitoring, stock trading, bioinformatics, medicine, and agriculture.

Table 2: Static Data Processing Vs Data Stream Processing

Parameter	Static Data	Data Stream
Data Schema	Static	Dynamic
Processing	Offline	Real-time
Data Generation Rate	Normal	Rapid
Storage of Data in Main Memory	Feasible	Not feasible
Computation Results	Accurate results are required	Approximate results are accepted
Data Access	Random	Sequential
Algorithms	1. Computation time is not a constraint 2. Multiple scans on data are possible 3. Memory is not a constraint	1. Computation time is most important as the data may loss 2. Single scan 3. Memory is a constraint (Limited memory)

Data stream clustering is the task of attaining valid and useful patterns out of a collection of d-dimensional data objects with memory and time constraints. In data stream clustering, data points are processed as an organized sequence, and it attracts much concentration in the recent years. The main issue in clustering is whether the data considered for mining is dynamic or static. Static data clustering is relatively much easier than dynamic data clustering that evolves over a period. In the case of static data set the entire data set is accessible for analysing and processing. But in the case of dynamic data, the entire data set is not available for analysing and processing. The comparison between the management of both static and dynamic data is shown in Table 2.

The domain of data stream clustering brings some important challenges to traditional data clustering because of their dynamic behaviour [4].

The following are the challenges which need to be addressed in data stream clustering.

- ***Handling evolving data.*** Data streams are potentially unbounded and they are evolved over a period. As the data stream develops, the behaviour of the data flow can be treated as a transition process over time, with new clusters may appear and the old clusters may disappear. The clustering technique must perform clustering incrementally to find evolving clusters over the time.
- ***Handling outlier data.*** In many cases, noisy data has a massive effect on the creation of clusters. Therefore, the clustering technique must handle the outliers in the data.
- ***Handling high-dimensional data***. Data streams are more quantitative. For example text data clustering is very popular in the social networks and web development. Thus, if the clustering technique is overwhelming in its data, this challenge must be overcome.
- ***Limited time.*** Data streams require a quick and real-time response. Therefore, the clustering technique is required to maintain the speed of the data stream in a limited time.
- ***Limited memory.*** Because of the huge amount of data, it is not possible to store the entire data in main memory for later analysis. The clustering technique should process and analyse the data stream in limited memory.
- ***One pass clustering.*** The data objects in the stream are read only once because of its high velocity. The clustering technique must process the data steam in single pass.

The main issue in data stream clustering is how to find the patterns from real-world data during single pass. Another main issue is how to store and process such a huge data in limited memory and limited time. Data stream clustering need the techniques which perform incrementally because it is not possible to store the entire data in main memory. In evolving approaches, streaming data can be captured through different window models [5] such as landmark window model (*LWM*), damped (fading) window model (*DWM*), and sliding window model (*SWM*).

2.1 Landmark Window Model (LWM)

Sometimes, it is very useful to process the evolution of streaming data beginning at a milestone or landmark. In this model, the window is represented by a specific time point called landmark and the present. This LMW is used to analyse the entire data stream from the landmark. In this LMW, all the data points have an equal weight 1. The main drawback of this LMW is that the most recent data points and the old data points have the same importance due to the equal weights. Note that the LMW is usually not much suitable for streaming data, as the amount of data inside the window increases to the high dimensions. Figure 2 exhibit the *LW* model.

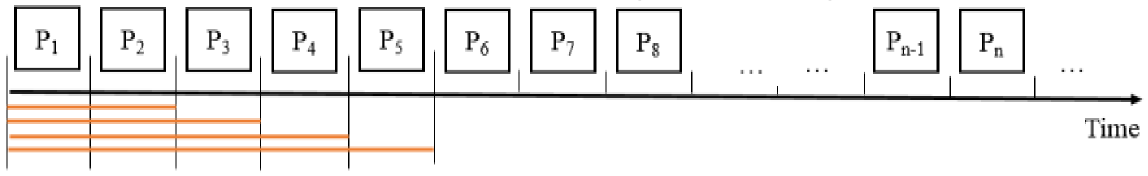

Figure 2: Landmark Window Model

2.2 Damped (fading) Window Model (DWM)

In many real-world domains, historical data elements are considered less useful than the more recent data elements. This model is adopted to minimize the impact of the historical data elements on the clustering results. In the *DWM*, We give a weight to each data element that will be decreased with time *t* via a fading/aging function *f(t)* [6]. According to this model, the more recent data elements have more weights than the historical data elements. The fading function *f(t)* is defined as *f(t)* = $2^{-\lambda t}$, where $\lambda > 0$ is a decay/fading parameter that represent the importance of historical data elements. If the λ value is high, then the importance of the historical data is low. Figure 3 exhibit the weights of data elements at various time stamps with the impact of fading factor λ. It is observed that the weight of a data element will decrease as its age increases.

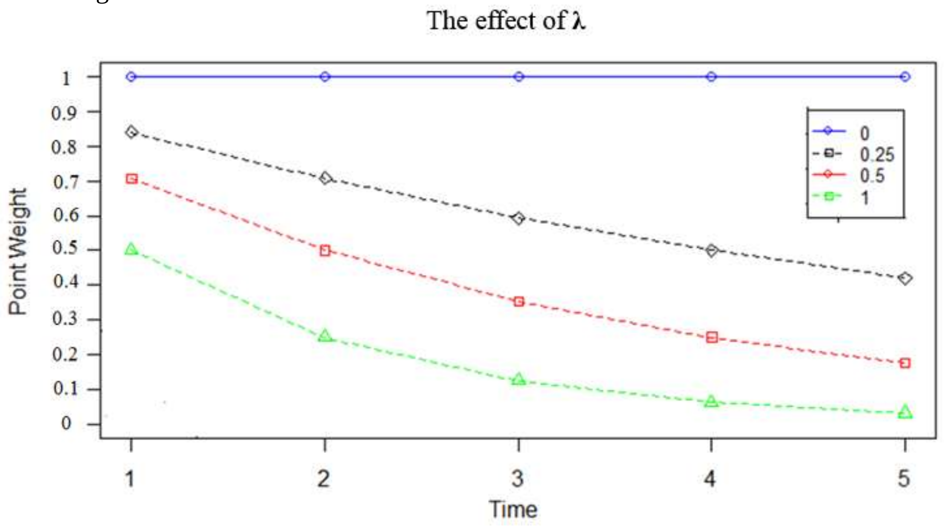

Figure 3: Damped Window Model

Table 3 shows the weights of data elements at various timestamps with various fading parameter values.

Table 3: Point Weights with the Effect of λ in Damped Window

Time	1	2	3	4	5
λ = 0	1	1	1	1	1
λ = 0.25	0.840	0.707	0.594	0.5	0.420
λ = 0.5	0.707	0.5	0.353	0.25	0.176
λ = 1	0.5	0.25	0.125	0.0625	0.03125

2.3 Sliding Window Model (SWM):

This model is an extensively used model for capturing and mining data streams. The main idea of this model is to adopt a fixed size window to capture the more recent data elements. In many cases, users are more interested only in discovering hidden patterns on most recent data elements, but not on the entire history of the data. Sliding window models are used to capture the most recent N objects. These windows only contain the recent *t* time units of data or the recent N data elements. In this SWM, all the date elements that are included within the size of an *SW* have a weight 1, and all the data elements that are not included within the size of an *SW* have a weight 0. The execution of this model is easy, but it will fail when the wrong window size is selected. Too narrow windows produce very accurate representations of the current state but are heavily influenced by noisy data, and due to the behavioural change of the streaming data too wide windows result in more stable and equally inaccurate results. Figure 4 shows the *SW* model.

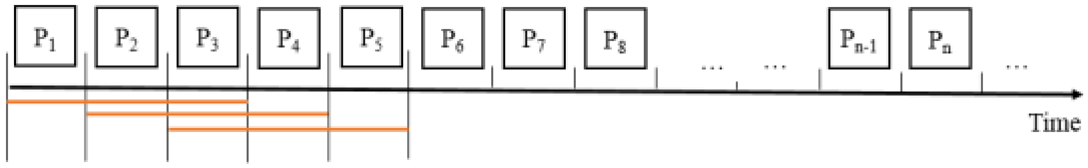

Figure 4: Sliding Window Model

In SWM, the summary statistics are computed for the streaming data that belongs to the timestamps between *(t-N + 1)* and *t*, where N is the size of an *SW* and *t* is the current timestamp.

3. DENSITY-BASED CLUSTERING FOR DATA STREAMS

Density-based clustering methods can be adopted to handle noisy data, and to discover the clusters with arbitrary shapes. DBSCAN [7], NBC [8], OPTICS [9], and DENCLUE [10] are some of the primary algorithms based on density. In the past decades, several techniques were proposed for clustering streaming data. Some of the techniques like C1uStream [1], HPStream [6], and SWClustering [11] are based on the enhanced K-means clustering technique that needs the value of k in advance. All of these methods have got good clustering results in dealing with streaming data, but they are failed to to discover arbitrary shape clusters. Density-based clustering techniques are more appropriate for clustering streaming data because they have the ability to find clusters of arbitrary shape and handle noisy data. Density-based clustering methods for streaming data are grouped into two broad categories called, 1) density-based micro-clustering methods and 2) density-grid-based clustering methods. The first category, i.e., Density-based micro-clustering methods includes various techniques like DenStream [12], OpticsStream [13], C-DenStream [14], rDenStream [15], SDStream[16], HDenStream [17], SOStream [18], HDDStream [19], PreDeCon Stream [20], FlockStream [21], LeaDenStream [22], DMMStream [23], DBStream [24]. The second category, i.e., Density-grid based clustering methods includes various techniques like DUCStream [25], DStream-I [26], DDStream [27], DStream-II [28], MR-Stream [29], PKS-Stream [30], DCUStream [31], DENGRIS-Stream [32], ExCC [33], GDRDD-Stream [34]. Several hybrid techniques such as HDC-Stream [35], RTDB-Stream [36], MCDAStream [37], StreamSW [38] have been proposed for clustering data streams. These hybrid techniques utilizes the advantages of both density-based micro-clustering methods and density grid-based methods.

Majority of the density-based clustering algorithms adopts a two-phase online-offline clustering framework [1] as shown in Figure 5. The online phase processes the streaming data in real-time and keeps the synopsis of the streaming data in the form of micro-clusters (*μCs*) or grids. The offline phase reclusters the *μCs* or grids into final macro-clusters. A variant of the classical clustering algorithm (e.g., DBSCAN [7]) is used in the offline phase for reclustering. The offline phase forms the final clusters based on the density within the *μCs* by considering the centers of *μCs* as pseudo points.

In many real-world domains, the users are more interested only in discovering hidden patterns in most recent data elements instead of the entire data stream. Here, sliding window (SW) models are generally used because they reflect the most recent N data points in the stream [5][6][11]. In sliding window model, the main idea is to use a fixed size sliding window in the data stream to capture the most recent data elements. In this model, all the data elements that are included within the size of an *SW* have a weight 1, and all the data elements that are not included within the size of an *SW* have a weight 0. In an *SW* model, the summary statistics are calculated for the stream data that belongs to the timestamps between *(t-N + 1)* and *t*, where N is the size of an *SW* and *t* is the current timestamp. The algorithms for dealing with all the data with equal importance will produce abnormal results because the bulk of processed data would not be accurate anymore. In this section we present different clustering algorithms for data streams over sliding window.

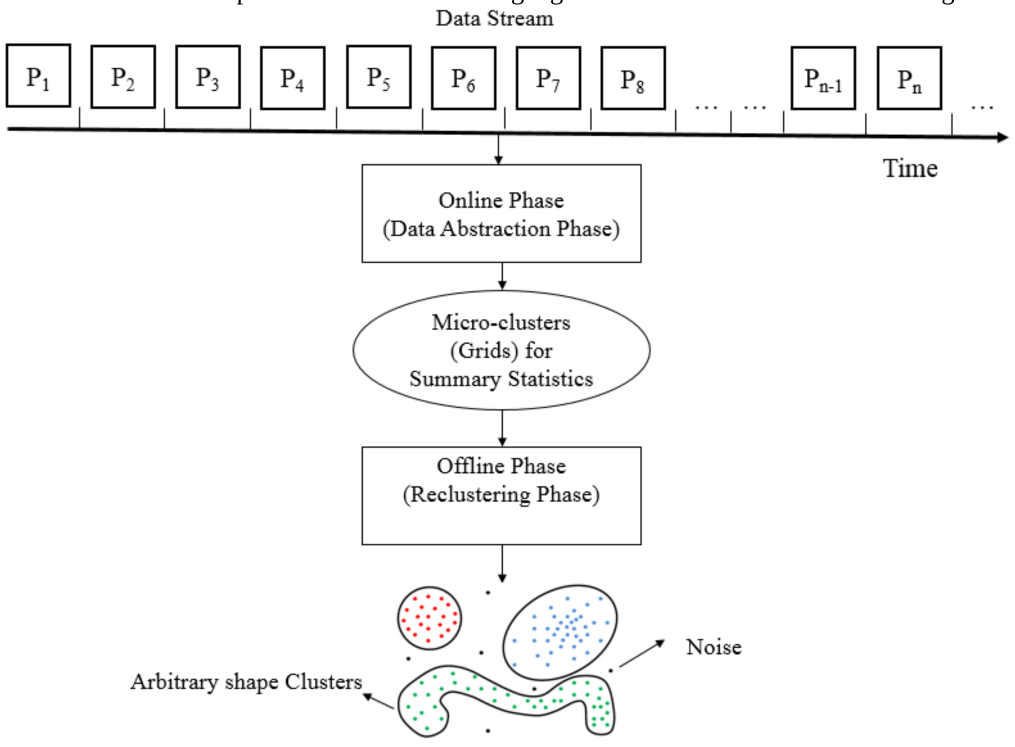

Figure 5: Two-phase Online-offline Clustering Framework

SDStream

Ren J et al. introduced an approach termed *SDStream* [16], which is an extension of SWClustering[11] for data stream clustering over sliding windows. The algorithm uses an online-offline framework of the CluStream [1] method. It considers the most recent N data elements and summarizes the historical data elemetns using sliding window model. The *SDStream* approach considers the distribution of most recent data elements, and it eliminates the data elemetns that are not included in the size of the sliding window. The algorithm uses the *potential-micro-cluster* and *outlier-micro-cluster* concepts which are similar to *DenStream*[12] to maintain actual clusters and outliers. The density of a *μC* is equal to the total number of data elements in it. The *potential-micro-clusters* and *outlier-micro-clusters* are stored in the form of *EHCF* [11]. Each EHCF represents a *potential-micro-cluster* or *outlier-micro-cluster*. The outdated data elements or *outlier-micro-clusters* which are not included in the bound of N-length sliding window are identified and eliminated via the value of *t* in TCF [11], where *t* is the timestamp of the most recent data element in the micor-cluster. In the offline phase, when a clustering request takes place, the algorithm executes a variant of DBSCAN [7] algorithm to discover the final clusters, using *potential-micro-clusters* as pseudo points. The *SDStream* algorithm can find arbitrary shape clusters over sliding windows, and it can handle noisy data.

DENGRIS-Stream

Most of the density grid-based algorithms uses a damped window model to maintain the streaming data. *Amineh et al.* introduced a two-phae density grid-based method termed *DENGRIS-Stream* [32] for clustering streaming data over sliding windows. In the online phase, the algorithm maps each new data element into a grid and computes the density of the gird for each sliding window. The offline phase clusters the grids based on the density of the grids. The offline phase modifies the final clusters in each sliding window by eliminating the expired and sparse grids, and combining the neighboring dense grids. The *DENGRIS-Stream* algorithm introduces the concept of expired grids to find the grids which are not active in the sliding window. For every grid, a feature vector is used in the online phase to keep the synopsis of the data elements in each grid. Therefore, the *DENGRIS-Stream* algorithm guarantees to capture the distribution of recent data elements objects precisely that is more acceptable in streaming data applications. The algorithm also maintains a grid-list which is modeled as a tree for the grids that are used for the clustering process. The tree structure allows to perform various operations like lookup, update, and delete quickly. This method can be treated as the first grid-based method for clustering streaming data over sliding windows. The method can find arbitrary shape clusters with limited time and memory.

STAGGER

Mohamed G. Elfeky *et al.* introduced an approach termed STAGGER[39] which is a single-pass, incremental, online clustering algorithm for mining periodic patterns in the streaming data. This method does not need that the user specific periodicity rate of the data in advance. Instead, it finds the potential periodicity rates. It also keeps multiple expanding sliding windows which are staggered over the data stream, where computations are shared among the multiple overlapping sliding windows. Thus, it is able to find a wide range of potential period lengths and also produce interactive output as well. Sliding windows that are in small length are imperative for early and real-time output, yet are limited to find short periodicity rates. As data streams arrives continuously, the sliding windows expand in length in order to cover the entire data stream. Larger-length sliding windows are able to find longer periodicity rates. STAGGER maintains a tree-like data structure incrementally for the frequent periodic patterns of each identified potential periodicity rate.

STAGGER identifies periodicity rates and finds frequent periodic patterns in single pass through the streaming data. This feature is very important when dealing with potentially unbounded data streams. To increase the coverage over the data stream, STAGGER adopts expanding sliding windows to identify all periodicity rates. The authors have proposed a novel method for handling the thresholds in order to enhance the accuracy of STAGGER. An empirical study using real and synthetic data validates the accuracy of STAGGER, shows the usefulness of the output, and shows the tradeoff between accuracy and processing time. Experimental results exhibit that STAGGER outperforms Fourier-Wavelet-based methods by an order of magnitude in terms of the accuracy of the discovered periodicity rates.

SWEM

Xuan Hong Dang et al. introduced an approach termed SWEM[40], that uses the Expectation Maximization Technique for data stream clustering in sliding windows. SWEM processes the streaming data in an incremental manner. It is also able to adapt to changes happened in the underlying streaming data distribution. SWEM contains three phases that are designed to address the problem of limited memory utilization and single-pass processing over streaming data. The SWEM method consists of two important operations. One is splitting, and another one is merging the micro components. These operations are used to automatically adapt to changes that are happened frequently in data stream distributions.

In the initial phase of SWEM, *m* distributions or micro components that are modelling the data elements within each time slot of the sliding window are computed. In this phase, it adopts the EM method to increase the average log likelihood measure. In the incremental phase, for the mixture models' parameters, SWEM uses the converged parameters in the previous time slot as the initial values. This is used to decrease the number of iterations if the stream's characteristic does not vary much. However, in case the stream's dispersion changes significantly, it is mandatory to relocate the components. Splitting and merging operations are developed in order to re-distribute the components discretely in the whole data space. Finally, the expiring phase is applied when the sliding window moves and the old time slot is expired from the mining model. SWEM algorithm is able to discover clusters with single pass over the data stream and works within limited memory. Importantly, two operations of splitting and merging the micro components were developed to address the problem of time-varying data streams. SWEM is designed based on the EM technique which is a mathematically sound tool. It has been shown to be stable and effective in many domains despite the mixture models it employs being assumed to follow Gaussian distributions.

StreamSW

StreamSW [38] algorithm uses a two-phase online-offline model for performing clustering on data streams over a sliding window. The online phase continuously reads a data element over a sliding window and adds it to an existing *potential-micro-cluster* or maps it to the grid cell. The algorithm uses the modified DBSCAN [7] algorithm in the offline phase to find the final macro clusters. Initially, a set of initial *potential-micro-clusters* are formed by the DBSCAN algorithm. The StreamSW algorithm has two components in its online phase: Merging and Mapping, and Pruning. In *Merging and Mapping*, when a new data element p arrives, the method for merging and mapping the point is as follows.

1. At first, the algorithm identifies the closest *potential-micro-cluster* c_p from *p*
2. Then the algorithm tries to merge p into its closest *potential-micro-cluster* c_p
3. If r_p, the newly calculated radius of c_p , is less than or equal to maximum radius ε, then *p* is merged into c_p.
4. Else, *p* is mapped into the grid *g*, and then the characteristic vector of the grid is updated.
5. If the grid *g* is a dense grid, then a new *potential-micro-cluster* is created from the grid *g*.
6. The associated grid *g* of the new *potential-micro-cluster* is deleted from the *grid_list*.

In the *Pruning Phase*, outlier clusters, sparse grids, and expired grids are discarded. The expired grids are identified and discarded based on their timestamp. If the timestamp of the grid is not in the sliding window length, then it is considered as an expired grid, and it will be removed from the *grid_list*. Outlier clusters and sparse grids are discarded from the gird list and *potential-micro-cluster* list, respectively, by checking their densities periodically.

The evaluation results exhibit that the StreamSW algorithm produces high-quality clusters and it has low execution time when compared to existing algorithms. It is observed that the StreamSW cannot handle high-dimensional data streams. The StreamSW requires high computation time and have lower performance on high-dimensional data because of the number of grid cells increases as the size of the space increases.

Table 4 shows the advantages and disadvantages of various clustering algorithms for data streams over sliding windows.

Table 4: Comparison of Clustering Algorithms

Algorithm	Advantages	Disadvantages
SDStream [16]	- Processes the most recent data and summarizes the old data by using a sliding window model. - It can handle noisy and evolving data.	- Cannot handle high-dimensional data. - The authors of SDStream did not clarify the main usage of the exponential histogram for their algorithm.
DENGRIS-Stream [32]	- First density clustering algorithm for evolving data streams over sliding window model.	- There is no evaluation to show the algorithm effectiveness compared with other algorithms.
STAGGER [39]	- STAGGER outperforms Fourier-Wavelet-based methods by an order of magnitude in terms of the accuracy of the discovered periodicity rates. - It uses the concept of expanding sliding window, and the most important advantage of having expanding sliding windows is, that is a much better average waiting time over a single sliding window.	- The memory consumption is more as it uses expanding sliding window. - Cannot handle high-dimensional data streams effectively -
SWEM[40]	- It is able to discover clusters with single pass over the data stream and works within limited memory. - It is designed based on the EM technique which is a mathematically sound tool	- Cannot handle high-dimensional data streams effectively
StreamSW [38]	- produces high-quality clusters and it has low execution time	- Cannot handle high-dimensional data streams - requires high computation time and have lower performance on high-dimensional data

4. EVALUATION OF ALGORITHMS AND CHALLENGING ISSUES

The data stream clustering domain poses many new challenges to classical clustering algorithms because of limited time and limited memory availability [41]. Table 5 shows various challenging issues and how the clusterig methods address those challenging issues in clustering data strems.

Table 5: Data Stream Clustering Algorithms and Challenging Issues

Algorithm	Outlier Data	Evolving Data	Limited Time	Limited Memory	High-Dimensional Data	Single Pass
SDStream [16]	Yes	Yes	No	Yes	No	Yes
DENGRIS-Stream [32]	Yes	Yes	No	Yes	No	Yes
STAGGER [39]	Yes	Yes	Yes	No	No	Yes
SWEM[40]	Yes	Yes	Yes	Yes	No	Yes
StreamSW [38]	Yes	Yes	Yes	Yes	No	Yes

5. CONCLUSION

In this paper, the most important algorithms for clustering data streams are reviewed and we have provided a comprehensive comparison between the different algorithms. Clustering data streams poses many challenges such as limited memory, limited time, handling evolving data, handling noisy data, and handling high-dimensional data. From the review of different methods for clustering data streams, it is observed that no existing clustering algorithm for data streams could handle all the challenging problems. Therefore, designing and developing an effective algorithm that can handle all the challenging issues is a challenging task in stream data clustering.

REFERENCES:

[1] Aggarwal C C, Wang J, Han J, et al. A framework for clustering evolving data streams[C]//Proceedings of the 29th international conference on Very large data bases-Volume 29. VLDB Endowment, 2003: 81-92.

[2] A. Jain, M. Murty, and P. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, 1999.

[3] Han J. and Kamber, M. "Data Mining Concepts and Techniques." 2nd Ed. Burlington: Morgan Kauffman, 2006.

[4] Guha S, Mishra N, Motwani R, O'Callaghan L. Clustering data streams. In Proc. the 41st Annual Symposium on Foundations of Computer Science, Nov. 2000, pp.359-366.

[5] Ng W, Dash M. Discovery of frequent patterns in transactional data streams. In *Lecture Notes in Computer Science 6380*, Hameurlain A, KÄung J, Wagner R *et al.* (eds.), Springer Berlin/Heidelberg, 2010, pp.1-30.

[6] Aggarwal C C, Han J, Wang J, Yu P S. A framework for projected clustering of high dimensional data streams. In *Proc. the 30th International Conference on Very Large Data Bases*, *Volume 30*, Aug. 29-Sept. 3, 2004, pp.852-863.

[7] Ester, M., Kriegel, H., Sander, J. & Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proc. of 2nd International Conference on Knowledge Discovery, 1996, pp. 226–231.

[8] Zhou S, Zhao Y, Guan J, & Huang J. A Neighborhood-Based Clustering Algorithm. In Pacific-Asia Conference on Knowledge Discovery and Data Mining 2005, pp 361-371.

[9] Ankerst, M., Breunig, M.M., Kriegel, H.-P. & Sander, J. OPTICS: Ordering Points to Identify the Clustering Structure. ACM SIGMOD Record, 1999, 28 (2). p.pp. 49–60.

[10] Hinneburg, A. & Keim, D.A. An Efficient Approach to Clustering in Large Multimedia Databases with Noise. In: KDD'98, 1998, New York, NY, pp. 58–65.

[11] Aoying Zhou, Feng Cao, Weining Qian, and Cheqing Jin. Tracking clusters in evolving data streams over sliding windows. Knowledge and Information Systems, 15:181–214, May 2008.

[12] Cao F, Ester M, Qian W, Zhou A. Density-Based Clustering Over an Evolving Data Stream with Noise. In Proc. the SIAM Conference on Data Mining, April 2006, pp.328-339.

[13] Tasoulis D K, Ross G, Adams N M. Visualising the Cluster Structure of Data Streams. In Proc. the 7th International Conference on Intelligent Data Analysis, Sept. 2007, pp.81- 92.

[14] Menasalvas E, Ruiz C, Spiliopoulou M. C-DenStream: Using Domain Knowledge on a Data Stream. In Proc. the 12th International Conference on Discovery Science, Oct. 2009, pp.287-301.

[15] Jing K, Liu L, Guo Y. A Three-Step Clustering Algorithm over an Evolving Data Stream. In Proc. the IEEE Int. Conf. Intelligent Computing and Intelligent Systems, Nov. 2009, pp.160-164.

[16] Ren J, Ma R. Density-Based Data Streams Clustering over Sliding Windows. In Proc. the 6th Int. Conf. Fuzzy Systems and Knowledge Discovery, Aug. 2009, pp.248-252

[17] Lin J, Lin H. A Density-Based Clustering over Evolving Heterogeneous Data Stream. In Proc. The 2nd Int. Colloquium on Computing, Communication, Control, and Management, Aug. 2009, pp.275-277.

[18] Dunham M, Isaksson C, Hahsler M. SOStream: Self Organizing Density-Based Clustering over Data Stream. In Lecture Notes in Computer Science 7376, Perner P (ed.), Springer Berlin Heidelberg, 2012, pp.264-278.

[19] Zimek A, Ntoutsi I, Palpanas T et al. Density-Based Projected Clustering over High Dimensional Data Streams. In Proc. The 12th SIAM Int. Conf. Data Mining, April 2012, pp.987-998.

[20] Spaus P, Hassani M, Gaber M M, Seidl T. Density-Based Projected Clustering of Data Streams. In Proc. the 6th Int. Conf. Scalable Uncertainty Management, Sept. 2012, pp.311-324.

[21] Pizzuti C, Forestiero A, Spezzano G. A Single Pass Algorithm for Clustering Evolving Data Streams based on Swarm Intelligence. Data Mining and Knowledge Discovery, 2013, 26(1): 1-26.

[22] Amineh A, Teh Ying W "LeaDen-Stream: A Leader Density-Based Clustering Algorithm over Evolving Data Stream." Journal of Computer and Communications, pp. 26-31, 2013.

[23] Amini A., Saboohi H., Wah T.Y., Herawan T. (2014) DMM-Stream: A Density Mini-Micro Clustering Algorithm for Evolving Data Streams. In: Herawan T., Deris M., Abawajy J. (eds) Proceedings of the First International Conference on Advanced Data and Information Engineering (DaEng-2013). Lecture Notes in Electrical Engineering, vol 285. Springer, Singapore.

[24] Hahsler M, and Matthew B. "Clustering Data Streams Based on Shared Density between Micro-Clusters." IEEE Transactions on Knowledge and Data Engineering, 2016.

[25] Li J, Gao J, Zhang Z, Tan P N. An incremental Data Stream Clustering Algorithm Based on Dense Units Detection. In Proc. the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, May 2005, pp.420-425.

[26] Chen Y, Tu L. Density-Based Clustering for Real-Time Stream Data. In Proc. the 13th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2007

[27] Tan C, Jia C, Yong A. A Grid and Density-Based Clustering Algorithm for Processing Data Stream. In Proc. the 2nd Int. Conf. Genetic and Evolutionary Computing, Sept. 2008, pp.517-521.

[28] Chen Y, Tu L, Stream Data Clustering Based on Grid Density and Attraction. ACM Transactions on Knowledge discovery Data, 2009, 3(3): Article No. 12.

[29] Ng W K, Wan L, Dang X H et al. Density-Based Clustering of Data Streams at Multiple Resolutions. ACM Trans. Knowledge Discovery from Data, 2009, 3(3).

[30] Ren J, Cai B, Hu C. Clustering over Data Streams Based on Grid Density and Index Tree. Journal of Convergence IT, 2011, 6(1): 83-93.

[31] Yang Y, Liu Z, Zhang J et al. Dynamic Density-Based Clustering Algorithm over Uncertain Data Streams. In Proc. the 9th Int. Conf. Fuzzy Systems and Knowledge Discovery, May 2012, pp.2664-2670.

[32] Amini A, Teh Ying W. DENGRIS-Stream: A Density-Grid Based Clustering Algorithm for Evolving Data Streams over Sliding Window. In Proc. International Conference on Data Mining and Computer Engineering, Dec. 2012, pp.206-210.

[33] Kaur S, Bhatnagar V, Chakravarthy S. Clustering Data Streams using Grid-Based Synopsis. Knowledge and Information Systems, June 2013.

[34] Zhang Y., Zhang J. A Density-Grid Based Clustering Algorithm on Data Stream Using Resilient Distributed Datasets. In: Khoury R., Drummond C. (eds) Advances in Artificial Intelligence. AI 2016. Lecture Notes in Computer Science, vol 9673. Springer, Cham.

[35] Amineh Amini, Hadi Saboohi, Teh YingWah, and Tutut Herawan. A Fast Density-Based Clustering Algorithm for Real-Time Internet of Things Stream, Hindawi Publishing Corporation, Volume 2014.

[36] K.S.S. Reddy, C.S. Bindu, RTDBSTREAM: A Real-time Density-based Clustering for Evolving Data Streams, Journal of Theoretical and Applied Information Technology, 2018

[37] K.S.S. Reddy, C.S. Bindu, MCDAStream: a real-time data stream clustering based on micro-cluster density and attraction, Int. J. Eng. Technol. 7 (2) (2018) 270–275.

[38] Reddy, K.S.S. and C.S. Bindu, StreamSW: A Densitybased Approach for Clustering Data Streams over Sliding Windows. Measurement, 2018.

[39] Elfeky, Mohamed G.; Aref, Walid G.; and Elmagarmid, Ahmed K., "STAGGER: Periodicity Mining of Data Streams using Expanding Sliding Windows" (2005). Department of Computer Science Technical Reports. Paper 1625. https://docs.lib.purdue.edu/cstech/1625.

[40] Dang, Xuan & Lee, Vincent & Ng, Wee Keong & Ciptadi, Arridhana & Ong, Kok-leong. (2009). An EM-Based Algorithm for Clustering Data Streams in Sliding Windows. 5463. 230-235. 10.1007/978-3-642-00887-0_18.

[41] K.S.S. Reddy, C. S. Bindu, A review on density-based clustering algorithms for big data analysis, in: International Conference on I-SMAC (IoT in Social, Mobile, Analytics, and Cloud) (I-SMAC) Palladam, Tamilnadu, India, 2017. pp. 123–130.