Int. J. Advance Soft Compu. Appl, Vol. 7, No. 3, November 2015
ISSN 2074-8523

Data Stream Clustering Algorithms: A Review

Maryam Mousavi[1], Azuraliza Abu Bakar[1], and Mohammadmahdi Vakilian[1]

[1]Centre for Artificial Intelligence Technology
Faculty of Information Science and Technology,
National University of Malaysia, 43600, Bangi, Selangor, Malaysia
e-mail: maryam.mousavi2010@gmail.com, azuraliza@ukm.edu.my,
mohammadmahdi.vakilian@gmail.com

Abstract

Data stream mining has become a research area of some interest in recent years. The key challenge in data stream mining is extracting valuable knowledge in real time from a massive, continuous, dynamic data stream in only a single scan. Clustering is an efficient tool to overcome this problem. Data stream clustering can be applied in various fields such as financial transactions, telephone records, sensor network monitoring, telecommunications, website analysis, weather monitoring, and e-business. Data stream clustering presents some challenges; it needs to be done in a short time frame with limited memory using a single-scan process. Moreover, because data stream outliers are hidden, clustering algorithms must be able to detect outliers and noise. In addition, the algorithms have to handle concept drift and detect arbitrary shaped clusters. Several algorithms have been proposed to overcome these challenges. This paper presents a review of five types of data stream clustering approaches: partitioning, hierarchical, density-based, grid-based and model-based. The different data stream clustering algorithms in the literature by considering their respective advantages and disadvantages are discussed.

Keywords: *Data Stream Clustering, Hierarchical Methods, Partitioning Methods, Grid-Based Methods, Density-Based Methods, Model-Based Methods.*

1 Introduction

The term data stream refers to a potentially bulky, continuous and fast sequence of information [1]. As opposed to traditional data forms which are unchanging and static, a data stream has its own unique characteristics: (i) it consists of a continuous flow of very large data; (ii) it is rapidly evolving data that occurs in real time with quick response requirements; (iii) multiple access to the data stream is almost impossible therefore algorithms have to be used to process it and are able to access the data once; (iv) storage of the data stream is restricted thus only a synopsis of the data can be saved and so finding the crucial data is a challenging task; and, (v) it is multidimensional therefore sophisticated algorithms are required to mine streaming data [2, 3]. Table 1 illustrates the main differences between data stream processing and traditional data processing [4]. Typical examples of streaming data include engineering data, scientific data, time series data, and data generated in other dynamic areas such as telephone records, sensor network monitoring, telecommunications, website analysis, weather monitoring, credit card, and e-business [5, 6].

The process of data stream mining involves extracting valuable patterns in real time from dynamic streaming data in only a single scan, which can be very challenging. However, the process of data stream clustering has been the subject of much attention due to its effectiveness in data mining. Clustering involves processing data and partitioning the information or objects contained within it into subsets known as clusters. The aim of this process is to classify similar objects into the same cluster while objects in various clusters are dissimilar [7]. The clustering process assists in restructuring the data by i) substituting a cluster with one or several new representatives, ii) classifying similar objects into groups, and iii) discovering patterns. Essentially, clustering algorithms that are used to process huge data are basic methods that can be applied in data mining, pattern recognition, and machine learning. Streaming access performs better than random access for the huge volumes of data stored on hard disks or in data stream form, hence streaming algorithms are required to cluster such data [8]. However, due to the nature of the data stream, which is massive and evolves over time, traditional clustering techniques cannot be applied. Thus, it has become crucial to develop new and improved clustering techniques.

The process of mining data streams by creating data clusters remains a challenge due to various factors: (i) single-scan clustering: data clustering has to be done quickly just once, in a single pass due to the data stream arriving continuously; (ii) limited time: data clusters have to be created in real time within a limited time frame; (iii) limited memory: the clustering algorithm is equipped with only limited memory but it has to process a continuous, incoming, infinite data stream; (iv) unknown number and shape of clusters: these aspects of the data stream remain unknown prior to processing; (v) evolving data: the algorithm has to be designed in such a way as to be prepared to handle the ever changing aspects of the data stream; and (vi) noisy data: noise in data affects clustering results so the clustering algorithm has to withstand the noise that exists in the data stream [9].

Table 1: Stream processing vs. traditional processing [4]

Stream processing	Traditional processing
Real-time processing	Offline processing
Rapid data generation relative to	Normal or slow data generation relative to

the available computational resources	the available computational resources
Storage of data is not feasible	Storage of data is feasible
Approximate results are acceptable	Accurate results are required
Processing of samples of data is the usual task	Processing of every data item/record is the usual task
Storage of aggregated and summarized data only	Storage of the raw data
Spatial and temporal contexts are particularly important	Spatial and temporal contexts are considered for certain classes of applications
Linear and sublinear computational techniques are widely used	Techniques with high space and time complexity are used if necessary

Recently, various perspectives on and aspects of data stream clustering have been discussed and several algorithms and methods have been proposed. The aim of this paper is to review this literature on data stream clustering algorithms. Reviews on this area have been published. In [2] a review of new and classic data stream clustering algorithms was conducted, while [10] presented a discussion of a comprehensive survey of 13 data stream clustering algorithms and their structures based on two categories (object-based and attribute-based). Also, another review of data stream clustering algorithms based on two different approaches, namely, clustering by example and clustering by variable has been presented [11]. However, in contrast to these previous works, this paper presents a review of five types of data stream clustering approaches: hierarchical, partitioning, grid-based, density-based and model-based.

2 Data Stream Clustering Methods

2.1 Hierarchical methods

A type of clustering techniques is hierarchical, which can be divided in two main types of methods, namely, agglomerative and divisive. The former type merges a set of ‘n’ objects into more general categories and the latter type divides ‘n’ objects into smaller clusters sequentially.

However, hierarchical agglomerative clustering (HAC) is more frequently used method with the option of manually determining the number of clusters [5]. The CURE [12] and the ROCK [13] algorithms are examples of the HAC algorithm that utilize a static model when selecting similar clusters that are to be integrated. Nevertheless, one drawback of such algorithms is that when a data point has been merged into a specific cluster, its membership is irrevocable. However, a cluster removal method has been developed that can overcome this limitation [14].

Unfortunately, this approach would be impractical for use with data streams because it requires multiple data scans [15].

BIRCH [16] was originally designed to mine traditional data. However, it has also been used to mine data streams due to its suitability for use with huge volumes of data, which gave rise to the micro and macroclustering concepts. These two concepts enable BIRCH to overcome two major drawbacks found in the HAC algorithm, namely, scalability and failure to undo what has been previously executed. This algorithm has two steps: in the first step, it scans the data base and then creates a tree consisting of information regarding data clusters. In the second step BIRCH prunes the tree by eliminating sparse nodes (outliers) and generating new original clusters. However, this method has a major drawback in the form of the limited capacity of its leaves. Moreover, this algorithm will not execute well if the clusters do not have spherical shapes because BIRCH controls the cluster's boundary by applying the notion of radius/diameter [17].

Online divisive agglomerative clustering (ODAC) is a time series data stream clustering technique [18, 19]. This algorithm is able to handle concept drift using both agglomerative and divisive hierarchical methods. It uses a top-down strategy to maintain a tree-like hierarchy of clusters. A correlation-based dissimilarity measure (splitting criterion) is utilized to split each node and then the agglomerative strategy is used to increase the detection of concept drift among the time series data.

The E-Stream algorithm [20], which is an evolution-based approach for clustering data streams, has also been proposed. This algorithm supports five kinds of evolution: (i) the appearance of a new cluster by agglomerating enough points in an area, (ii) the disappearance of existing clusters by considering the fading structure, (iii) the evolution of a cluster by changing the behaviour of data, (iv) the merging of a pair of similar clusters, and (v) the splitting of a cluster into two subclusters. However, the E-Stream has a polynomial runtime ($O(k^2)$) with regard to the number of clusters in the merging process.

An extension of the E-Stream algorithm known as HUE-Stream [21] has been proposed which is an evolution-based method for supporting uncertainty in heterogeneous streaming data. A distance function with probability distribution of two objects is presented to handle uncertainty in categorical attributes. For change detection in the clustering structure, the proposed distance function is utilized for merging clusters and finding the nearest cluster of the given new incoming data and the proposed histogram management is used for splitting cluster in categorical data. Experiments were conducted to compare the results of HUE-Stream with those of UMicro [22], which was an algorithm for clustering uncertain data streams and it was found that HUE-Stream outperformed UMicro in terms of cluster quality but it needed more parameters to be set by user.

2.2 Partitioning methods

Some data stream clustering techniques are based on partitioning techniques such as *k*-median and *k*-means [23]. A *k*-median-based clustering algorithm, the StreamLSearch algorithm [24] has been proposed for clustering high quality data streams. It is a two-part sequence starting with the determination of sample size by the STREAM algorithm. Then, if the size of the sample is larger than the outcome determined from a predefined equation, the LSEARCH algorithm is applied. Every data chunk is similarly processed and then the LSEARCH is applied to the cluster centres that have been created. This algorithm was found to be superior to BIRCH [16] in terms of the sum of squared distance.

An incremental *k*-means algorithm to create binary data stream clusters was proposed [25]. Several experiments have demonstrated that this modified algorithm is far better than the scalable *k*-means approach. The advantages of utilizing binary data are that it facilitates the manipulation of categorical data and disregards data normalization. This algorithm updates the centre and weight of each cluster after inspecting a number of transactions, which balances the square root of the number of transactions as opposed to updating them one by one.

Another partitioning method, CluStream [26], is a two-component clustering method that clusters data using an online microclustering and an offline macroclustering component. The first step involves acquiring summary statistics from the data stream, which is completed by the online microclustering component. Then, the second component utilizes these statistics as well as other inputs to create clusters. However, the *k*-means algorithm embedded in the offline macroclustering component of this two-component method has a number of drawbacks, the main one being the inability of the algorithm to detect arbitrary shaped clusters. The *k*-means focuses more on detecting spherical clusters even though non-convex and interwoven clusters are also used in various applications. This algorithm is also incapable of detecting noise and outliers. It is also unsuitable for use with large data streams because it needs multiple scans of the data. Thus, due to this drawback, the CluStream has to compress raw data streams into microclusters via an online process and then use these microclusters in its offline phase**.**

HPStream [27] was developed as an extension to CluStream**.** This algorithm is a projected clustering for high-dimensional streaming data. The primary motivation behind this extension is that CluStream does not perform efficiently when applied to a high-dimensional data stream. However, although the HPStream algorithm is capable of handling high-dimensional cases, it is difficult to obtain an appropriate average projection dimension.

The SWClustering algorithm [28] is capable of identifying clusters in data streams over the sliding window model. A new data structure known as the exponential histogram cluster feature (EHCF) is introduced by this algorithm and it is capable of capturing in-cluster evolution. This algorithm keeps aggregates over the sliding window model and it is capable of calculating clusters based on the synopses that the EHCF provides. Nevertheless, SWClustering was developed according to the *k*-means algorithm [23], thus it is unable to determine arbitrary shaped clusters as well as being incapable of handling outliers.

STREAMKM++ [8] is a type of *k*-means algorithm that is suitable for clustering data streams from a Euclidean space. If the number of cluster centres is large, the quality of the results derived from this algorithm is better than those by BIRCH and StreamLSearch, but in terms of running time this algorithm is slower than BIRCH.

2.3 Grid-based methods

Grid-based clustering algorithms such as CLIQUE [29], WaveCluster [30] and STING [31] have a very unique characteristic in that their processing time is not dependent on the number of data points, which makes them fast. These algorithms utilize a multi-resolution grid structure. This structure separates an object's space into a predetermined number of cells. Then, these cells form the grid structure where all clustering processes are conducted.

GCHDS is a grid-based clustering algorithm to cluster high-dimensional data streams [32]. It fulfils the three requirements for online data stream clustering; a single scan over the data, high-speed processing and limited memory usage. In this algorithm, a grid structure is utilized to create a synopsis of the online data stream. The time to maintain the grid structure is too short which it can be neglected when the grid has been enlarged widely enough to contain most of the data in the data stream. By analysing the data distribution on each dimension, useful dimensions are selected to construct a subspace in which the clustering process is performed. Experiments showed that the GCHDS algorithm has high clustering accuracy when the parameters are properly set. It outperforms the HPStream algorithm, which is also a subspace clustering algorithm for high-dimensional data streams, in terms of clustering accuracy. However, the GCHDS algorithm can find only clusters that belong to the same subspace, whereas in real data sets, the clusters can belong to various subspaces. To solve this problem, the Grid-based Subspace Clustering algorithm for high-dimensional Data Streams (GSCDS) has been proposed which can detect clusters in various subspaces [33]. To create a synopsis of the data stream, this algorithm utilizes a grid data structure which has been partitioned uniformly. After that, to find the subspaces which consist of clusters, the top-down grid-based technique is applied. Then, to detect clusters in every subspace, GSCDS applies the bottom-up grid-based technique. The experimental results demonstrate that GSCDS outperforms GCHDS in terms of clustering quality.

A distributed grid clustering algorithm known as DGClust has been proposed for data streams generated in sensor networks [34]. It uses the grid structure to summarize the data stream. This algorithm allows every local sensor to retain the online discretization of its streaming data and it performs with a fixed update time and space to reduce dimensionality and the communication burden.

2.4 Density-based methods

Density-based algorithms possess quite a few significant advantages for data clustering such as i) the ability to detect arbitrary shaped clusters, ii) the ability to handle noise and iii) they require just the one time to scan raw data. Apart from that, such algorithms do not require prior knowledge of the number of clusters (*k*) unlike *k*-means algorithms that need to be given the number of clusters in advance [35-37].

DBSCAN [38], GDBSCAN [39] and DENCLUE [40] are all density-based clustering algorithms that can be used to detect any arbitrary shaped clusters. However, they are unsuitable for processing clusters in data streams. An extension of the DBSCAN known as the incremental-DBSCAN [41] was developed. This method can proficiently add and remove points incrementally in data warehousing. It is capable of detecting arbitrary shaped clusters but requires parameters tuning.

For static data sets, the OPTICS algorithm is the solution for density-based clustering algorithms that are dependent on parameters [42]. It contains two concepts for organizing points: i) the core distance and ii) the reachability distance. In the clustering process, the reachability distance and spatial positioning order the organized points to be added to the clustering structure list. It includes a comprehensive parameter setting for a single clustering structure. Unfortunately, the OPTICS is not suitable for use in data streams although it is perfect for parameter-dependent problems and is capable of detecting overlapping clusters and arbitrary shaped clusters.

Another improvement of the DBSCAN algorithm known as LDBSCAN [43] has also been proposed. This algorithm uses the concept of local density-based clustering. It is able to detect density-based local outliers and noise. However, this algorithm does not work well in data streams.

A two-phase scheme density-based algorithm known as DenStream has been developed to cluster evolving data streams [44]. In the first phase, this algorithm uses the fading window model to create a synopsis of the data. Then, in the second phase, the synopsis of the data stored from first phase is utilized to provide the clustering result. This algorithm can handle arbitrary shaped clusters, but due to the numerous time vector calculations, it has high time complexity [45].

An improvement of the DenStream algorithm is rDenStream [46], which is a three-phase clustering algorithm. In this algorithm, previously discarded unimportant clusters are stored in a transitory memory. This approach ensures that this data has the chance to form clusters and increase the clustering accuracy. rDenStream can handle a huge number of outliers and its first two phases are comparable to those of DenStream but it has an additional phase known as the retrospect. This phase allows the algorithm to learn from the discarded data to increase its accuracy. From an experimental comparison, rDenStream outperforms DenStream in the initial phase. However, this algorithm requires more time and memory as compared to DenStream because it processes and saves the historical buffer.

The D-Stream algorithm [35] has also been proposed, which is capable of making automatic and dynamic adjustments to the data clusters without user specification with regard to the target time horizon and number of clusters**.** This algorithm creates separated grids to map new incoming data. A decay factor is used with the density of each data point in order to determine which data are recent and which are less important (old). The D-Stream algorithm is incapable of processing very high-dimensional data; however, the DenStream algorithm has no difficulty in processing such data. Additionally, D-Stream and DenStream have been found to outperform CluStream.

Similar to D-Stream, MR-Stream [47] creates cell partitions in the data space. Whenever a dimension is divided in half, a single cell goes through another division to form 2^d subcells, where d is the dimension of the data set. The division process can be set to a maximum limit by a user-defined parameter. The divided cells are stored on a quad tree structure that allows for data clusters to be created at different resolution levels. The MR-Stream algorithm allocates all new data into the appropriate cells at every time stamp interval during the online phase and also updates the summarized data. In a comparison between MR-Stream and D-Stream, MR-Stream showed better performance.

Another density-based clustering algorithm for streaming data is the DSCLU algorithm [48]. DSCLU uses microclusters to detect suitable clusters, focusing on localizing dominant microclusters on the basis of their neighbours' weight. It is able to detect clusters in multi-density environments.

OPCluStream is another density-based algorithm for clustering data streams [49]. This algorithm utilizes a tree topology for organizing points and directional pointers to link all related points together**.** This algorithm is able to detect arbitrary and overlapping clusters.

2.5 Model-based methods

Another type of clustering is the model-based method that runs a hypothesized model for every cluster and determines which data will fit the model perfectly. The COBWEB algorithm [50] is one such model-based algorithm and it is an incremental conceptual method to cluster data. This method uses the tree structure generated by a category function. It generates a hierarchical clustering in the form of a classification tree. In this form, each node keeps a notion and has a probabilistic description of that notion which summarizes the objects classified under the nodes. COBWEB can detect outliers but because it utilizes the tree structure, it has a limitation in terms of the capacity of the leaves [51].

CluDistream [52] is an algorithm that has been developed based on the expectation maximization technique for clustering streaming data. This algorithm is only capable of dealing with the clustering problem in a landmark window with the expectation maximization executed at every node of the distributed network. Nonetheless, CluDistream has been found to have significant results, particularly when it is implemented in distributed stream environments where transmitted data can either be noisy or missing.

The SWEM algorithm [53] clusters data streams in a time-based sliding window with the expectation maximization technique. This algorithm consists of two phases; in the first phase by scanning the data, it creates a synopsis of that data as microcomponents. After that, in the second phase this data synopsis is utilized to create global data clusters. This two-step structure is designed to deal with the limited memory and single-scan processing problems of the data stream. This algorithm is able to detect noise and handle the missing data properly. SWEM when compared to the CluStream algorithm was found to show better performance in terms of time complexity and quality of clusters.

3 Discussion

Beside the algorithms mentioned in the previous section, there are some other approaches for clustering data streams such as [54-56]. In [54] an algorithm was proposed which is based on artificial immune systems and is known as TECNO-STREAMS. This algorithm is capable of identifying an unknown number of clusters in a data stream with noise. In this algorithm, multiple B-cells can show a single cluster so this algorithm can detect arbitrary shaped clusters; however, it is unable to solve high-dimensional cases well.

In [55, 56] the authors used a bio-inspired model known as the flocking model to cluster a data stream. In [55] the multiple species flocking (MSF) model was proposed for the clustering of streaming documents. The advantage of this model is that it uses a heuristic mechanism to search for flocks in the virtual space. Agents move according to MSF rules into the space and when they encounter other agents in a predefined visibility range, they can decide to form a flock if they are similar. Flocks can join to form swarms of similar groups, where a swarm represents a cluster. In [56] FADS was proposed, which is a multi-agent algorithm to detect anomalies in a data stream. This method is applicable for very large data sets.

All the approaches highlighted in this paper have their advantages and disadvantages as shown in Table 2. They perform the clustering process by focusing on different aspects. For instance, a number of them place an emphasis on the need to handle noises and outliers, whereas others neglect this aspect.

Some of these algorithms are more accurate than others but they often have the drawback of high time complexity. Some of them use the whole data stream to create data clusters, but others just use a synopsis of the data stream. Therefore, currently there is no algorithm that offers the best performance in terms of all the necessary features such as high quality, low computational process, noise detection, etc. Hence, as yet we are limited to choosing an algorithm that best fits our purpose.

Table 2: Advantages and disadvantages of clustering methods

Method	Advantages	Disadvantages
Partitioning	Easy to implement The use of iterative way to create the clusters	The number of clusters should be predefined by user Only spherical shaped clusters can be determined
Hierarchical	Easy to handle any forms of similarity or distance	Ambiguity of termination criteria High complexity
Grid-based	Fast processing time Can handle noises	Can not apply to high dimensional data The size of grid should be predefined
Density-based	Can detect arbitrary shaped clusters Can handle noises	Several parameters are needed to be provided in advance Does not work well in multi-density data
Model-based	Specifying the number of clusters automatically based on standard statistics Can handle noises	Depending on the hypothesized model or structure

4 Conclusion and Future Work

The major research field in data stream mining is to develop efficient methods to mine the data stream. However, the mining task is complicated because of the specific characteristics of the data stream; it is massive, even potentially infinite, and is, moreover, continuous, requires a single scan, and dynamically changes over the time, thus requiring a rapid response usually in real time. The data stream clustering approach is one of the data mining techniques that can extract knowledge from such data. Conventional clustering methods are not flexible enough to tackle evolving data. Hence, in recent years, the demand for efficient data clustering algorithms has led to the publication of numerous methods.

This paper has presented a review of five types of clustering methods that have emerged in the field of data stream clustering. In practice, each algorithm can be useful based on its applications and properties. In future work, we aim to develop and implement an efficient data stream clustering algorithm to overcome the drawbacks of previous data stream clustering approaches.

ACKNOWLEDGEMENTS
This work is supported by Ministry of Higher Education Malaysia under projects “FRGS/1/2013/ICT02/UKM/01/1” and “ERGS/1/2012/STG07/UKM/01/1”.

References

[1] Ericsson, "5G for the networked society beyond 2020," *Mobile World Congress 2013,* February 2013.

[1] R. Mythily, A. Banu, and S. Raghunathan, "Clustering Models for Data Stream Mining," *Procedia Computer Science,* vol. 46, pp. 619-626, 2015.

[2] S. Ding, F. Wu, J. Qian, H. Jia, and F. Jin, "Research on data stream clustering algorithms," *Artificial Intelligence Review,* pp. 1-8, 2013.

[3] Y.-H. Lu and Y. Huang, "Mining data streams using clustering," in *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, 2005, pp. 2079-2083.

[4] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "Data stream mining," in *Data Mining and Knowledge Discovery Handbook*, ed: Springer, 2010, pp. 759-787.

[5] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*: Morgan kaufmann, 2006.

[6] H. Yang, D. Yi, and C. Yu, "Cluster Data Streams with Noisy Variables," *Communications in Statistics-Simulation and Computation,* pp. 00-00, 2014.

[7] A. Madraky, Z. A. Othman, and A. R. Hamdan, "Analytic Methods for Spatio-Temporal Data in a Nature-Inspired Data Model," *International Review on Computers and Software (IRECOS),* vol. 9, pp. 547-556, 2014.

[8] M. R. Ackermann, M. Märtens, C. Raupach, K. Swierkot, C. Lammersen, and C. Sohler, "StreamKM++: A clustering algorithm for data streams," *Journal of Experimental Algorithmics (JEA),* vol. 17, p. 2.4, 2012.

[9] L. Xu and J. Xun, "Research on Distributed Data Stream Mining in Internet of Things," in *International Conference on Logistics Engineering, Management and Computer Science (LEMCS 2014)*, 2014.

[10]J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. d. Carvalho, and J. Gama, "Data stream clustering: A survey," *ACM Computing Surveys (CSUR),* vol. 46, p. 13, 2013.

[11]D. Toshniwal, "Clustering techniques for streaming data-a survey," in *Advance Computing Conference (IACC), 2013 IEEE 3rd International*, 2013, pp. 951-956.

[12]S. Guha, R. Rastogi, and K. Shim, "CURE: an efficient clustering algorithm for large databases," in *ACM SIGMOD Record*, 1998, pp. 73-84.

[13]S. Guha, R. Rastogi, and K. Shim, "ROCK: A robust clustering algorithm for categorical attributes," *Information systems,* vol. 25, pp. 345-366, 2000.

[14]P. Fränti and O. Virmajoki, "Iterative shrinking method for clustering problems," *Pattern Recognition,* vol. 39, pp. 761-775, 2006.

[15]Q. Tu, J. Lu, B. Yuan, J. Tang, and J.-Y. Yang, "Density-based hierarchical clustering for streaming data," *Pattern Recognition Letters,* vol. 33, pp. 641-645, 2012.

[16]T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an efficient data clustering method for very large databases," in *ACM SIGMOD Record*, 1996, pp. 103-114.

[17]M. Khalilian, N. Mustapha, M. N. Sulaiman, and A. Mamat, "Different Aspects of Data Stream Clustering," in *Innovations and Advances in Computer, Information, Systems Sciences, and Engineering*, ed: Springer, 2013, pp. 1181-1191.

[18]P. P. Rodrigues, J. Gama, and J. P. Pedroso, "Hierarchical clustering of time-series data streams," *Knowledge and Data Engineering, IEEE Transactions on,* vol. 20, pp. 615-627, 2008.

[19]P. P. Rodrigues, J. Gama, and J. P. Pedroso, "ODAC: Hierarchical Clustering of Time Series Data Streams," in *SDM*, 2006.

[20]K. Udommanetanakit, T. Rakthanmanon, and K. Waiyamai, "E-stream: Evolution-based technique for stream clustering," in *Advanced Data Mining and Applications*, ed: Springer, 2007, pp. 605-615.

[21]W. Meesuksabai, T. Kangkachit, and K. Waiyamai, "HUE-Stream: evolution-based clustering technique for heterogeneous data streams with uncertainty," in *Advanced Data Mining and Applications*, ed: Springer, 2011, pp. 27-40.

[22]C. C. Aggarwal and P. S. Yu, "A framework for clustering uncertain data streams," in *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, 2008, pp. 150-159.

[23]A. K. Jain and R. C. Dubes, *Algorithms for clustering data*: Prentice-Hall, Inc., 1988.

[24]L. O'callaghan, N. Mishra, A. Meyerson, S. Guha, and R. Motwani, "Streaming-data algorithms for high-quality clustering," in *Data Engineering, 2002. Proceedings. 18th International Conference on*, 2002, pp. 685-694.

[25]C. Ordonez, "Clustering binary data streams with K-means," in *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, 2003, pp. 12-19.

[26]C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "A framework for clustering evolving data streams," in *Proceedings of the 29th international conference on Very large data bases-Volume 29*, 2003, pp. 81-92.

[27]C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "A framework for projected clustering of high dimensional data streams," in *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, 2004, pp. 852-863.

[28]A. Zhou, F. Cao, W. Qian, and C. Jin, "Tracking clusters in evolving data streams over sliding windows," *Knowledge and Information Systems,* vol. 15, pp. 181-214, 2008.

[29]R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, *Automatic subspace clustering of high dimensional data for data mining applications* vol. 27: ACM, 1998.

[30]G. Sheikholeslami, S. Chatterjee, and A. Zhang, "WaveCluster: a wavelet-based clustering approach for spatial data in very large databases," *The VLDB Journal,* vol. 8, pp. 289-304, 2000.

[31]W. Wang, J. Yang, and R. Muntz, "STING: A statistical information grid approach to spatial data mining," in *VLDB*, 1997, pp. 186-195.

[32]Y. Lu, Y. Sun, G. Xu, and G. Liu, "A grid-based clustering algorithm for high-dimensional data streams," in *Advanced Data Mining and Applications*, ed: Springer, 2005, pp. 824-831.

[33]Y. Sun and Y. Lu, "A grid-based subspace clustering algorithm for high-dimensional data streams," in *Web Information Systems–WISE 2006 Workshops*, 2006, pp. 37-48.

[34]J. Gama, P. P. Rodrigues, and L. Lopes, "Clustering distributed sensor data streams using local processing and reduced communication," *Intelligent Data Analysis,* vol. 15, pp. 3-28, 2011.

[35]L. Tu and Y. Chen, "Stream data clustering based on grid density and attraction," *ACM Transactions on Knowledge Discovery from Data (TKDD),* vol. 3, p. 12, 2009.

[36]H.-L. Nguyen, Y.-K. Woon, and W.-K. Ng, "A survey on data stream clustering and classification," *Knowledge and Information Systems,* pp. 1-35, 2014.

[37]W.-K. Loh and Y.-H. Park, "A Survey on Density-Based Clustering Algorithms," in *Ubiquitous Information Technologies and Applications*, ed: Springer, 2014, pp. 775-780.

[38]M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," 1996.

[39]J. Sander, M. Ester, H.-P. Kriegel, and X. Xu, "Density-based clustering in spatial databases: The algorithm gdbscan and its applications," *Data Mining and Knowledge Discovery,* vol. 2, pp. 169-194, 1998.

[40]A. Hinneburg and D. A. Keim, *An efficient approach to clustering in large multimedia databases with noise*: Bibliothek der Universität Konstanz, 1998.

[41]M. Ester, H.-P. Kriegel, J. Sander, M. Wimmer, and X. Xu, "Incremental clustering for mining in a data warehousing environment," in *Proceedings of the International Conference on Very Large Data Bases*, 1998, pp. 323-333.

[42]M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "OPTICS: ordering points to identify the clustering structure," *ACM SIGMOD Record,* vol. 28, pp. 49-60, 1999.

[43]L. Duan, L. Xu, F. Guo, J. Lee, and B. Yan, "A local-density based spatial clustering algorithm with noise," *Information Systems,* vol. 32, pp. 978-986, 2007.

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

[45]A. Forestiero, C. Pizzuti, and G. Spezzano, "A single pass algorithm for clustering evolving data streams based on swarm intelligence," *Data Mining and Knowledge Discovery,* vol. 26, pp. 1-26, 2013.

[46]L. Li-xiong, K. Jing, G. Yun-fei, and H. Hai, "A three-step clustering algorithm over an evolving data stream," in *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on*, 2009, pp. 160-164.

[47]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 (TKDD),* vol. 3, p. 14, 2009.

[48]A. Namadchian and G. Esfandani, "DSCLU: a new Data Stream CLUstring algorithm for multi density environments," in *Software Engineering, Artificial Intelligence, Networking and Parallel & Distributed Computing (SNPD), 2012 13th ACIS International Conference on*, 2012, pp. 83-88.

[49]H. Wang, Y. Yu, Q. Wang, and Y. Wan, "A density-based clustering structure mining algorithm for data streams," in *Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*, 2012, pp. 69-76.

[50]D. Fisher, "Iterative optimization and simplification of hierarchical clusterings," *arXiv preprint cs/9604103,* 1996.

[51]M. Khalilian and N. Mustapha, "Data stream clustering: Challenges and issues," *arXiv preprint arXiv:1006.5261,* 2010.

[52]A. Zhou, F. Cao, Y. Yan, C. Sha, and X. He, "Distributed data stream clustering: A fast EM-based approach," in *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, 2007, pp. 736-745.

[53]X. H. Dang, V. C. Lee, W. K. Ng, and K. L. Ong, "Incremental and adaptive clustering stream data over sliding window," in *Database and Expert Systems Applications*, 2009, pp. 660-674.

[54]O. Nasraoui, C. C. Uribe, C. R. Coronel, and F. Gonzalez, "Tecno-streams: tracking evolving clusters in noisy data streams with a scalable immune system learning model," in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, 2003, pp. 235-242.

[55]X. Cui and T. E. Potok, "A distributed agent implementation of multiple species flocking model for document partitioning clustering," in *Cooperative Information Agents X*, ed: Springer, 2006, pp. 124-137.

[56]A. Forestiero, "FADS: Flocking anomalies in data streams," in *Intelligent Systems (IS), 2012 6th IEEE International Conference*, 2012, pp. 461-466.