Volume 7, Issue 1, January 2017 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Survey on Density based Micro-clustering Algorithms for Data Stream Clustering

Donia Augustine*
Department of CSE, Thejus Engineering College, Vellarakkad,
Kerala, India

DOI:: 10.23956/ijarcsse/V7I1/0111

Abstract— As applications producing streaming data has increased tremendously, data stream mining became a necessity. 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. 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 such algorithms which clusters the data stream using the density estimate. The different data stream clustering algorithms in the literature are reviewed by considering their respective advantages and disadvantages and are compared based on several parameters.

Keywords— Datastream, Micro-cluster, Outlier, DenStream, HDDStream, LeaDenStream

I. INTRODUCTION

Typical statistical and data mining methods (e.g., clustering, regression, classification and frequent pattern mining) work with static data sets, meaning that the complete data set is available as a whole to perform all necessary computations. Well known methods like k- means clustering, linear regression, decision tree induction and the APRIORI algorithm to find frequent itemsets scan the complete data set repeatedly to produce their results. However, in recent years more and more applications need to work with data which are not static, but are the result of a continuous data generating process which is likely to evolve over time. Some examples are web click-stream data, computer network monitoring data, telecommunication connection data, readings from sensor nets and stock quotes. These types of data are called data streams and dealing with data streams has become an increasingly important area of research. Early on, the statistics community also recognized the importance of the emerging field of statistical analysis of massive data streams.

Clustering is a significant data stream mining task. However clustering in data stream environments need some special requirements due to the data stream characteristics such as clustering in limited memory and time with single pass over the evolving data streams and further handling noisy data. The term data stream refers to a potentially bulky, continuous and fast sequence of information. 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 I 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 ebusiness [5,6].

Mining data stream became more prominent in many applications, including real-time detection of anomalies in computer network traffic, web searches, monitoring environmental sensors, social networks, sensor networks, and cyber physical systems. In these applications, data streams arrive continuously and evolve significantly over time. Mining data streams is related to extracting knowledge structure represented in streams information. 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].

Table I Traditional Processing Vs Stream Processing

Traditional Processing	**Stream Processing**
Offline processing	Real time processing
Normal or slow data generation	Rapid data generation
Storage of data is feasible	Storage of data is not feasible
Accurate results are required	Approximate results are acceptable
Need processing of every dataitem	Need processing of sample data
Storage of raw data	Storage of summarized data
Application dependent contexts	Consider special and temporal contexts

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].

II. DENSITY BASED MICRO-CLUSTERING

In this section, we will introduce and analyze the outstanding density-based clustering algorithm based on microclusters. One of the well-known designs for clustering data stream is two-phase clustering, which Aggarwal et al. introduce. The two-phase clustering separates the clustering process into online and offline components. In this onlineoffline way, the online phase captures synopsis information from the data stream, and the offline phase generates clusters on the stored synopsis information. Density-based clustering has the ability to discover clusters in any shape. It defines the clusters by separating dense area from sparse ones. Among the density-based algorithms that are explained earlier in this paper, DBSCAN is used in the offline phase for clustering algorithms on data streams. In clustering data streams, it is impractical to save all the incoming data objects. Micro-clusters are a popular technique in stream clustering, which maintain the compact representation of the clustering.

A. DenStream Algorithm

This algorithm has ability to handle noise. This algorithm use fading window model for clustering the stream data. The algorithm expands the micro-cluster concept as core micro-cluster, potential micro-cluster, and outlier micro-cluster in order to distinguish real data and outliers [3]. It is based on the online-offline framework.

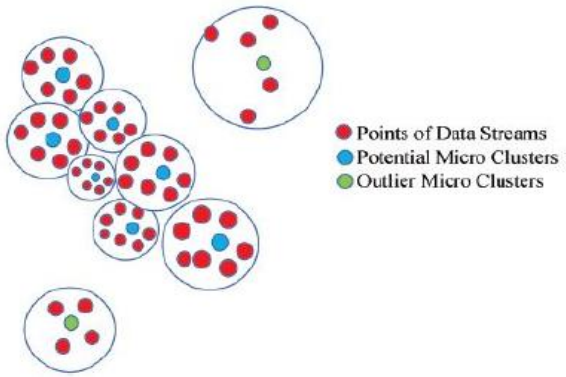

Fig.1: Potential and Outlier Microclusters

Online Phase: For initialization of the online phase, DenStream uses the DBSCAN algorithm on the first initial points, and forms the initial potential microclusters. In fact, for each data point, if the aggregate of the weights of the data points in the neighborhood radius is above the weight threshold, then a potential micro-cluster is created. When a new data point arrives, it is added to either the nearest existing potential micro-cluster or outlier micro-cluster. The Euclidean distance between the new data point and the center of the nearest potential or outlier micro-cluster is measured. A micro-cluster is chosen with the distance less than or equal to the radius threshold. If it does not belong to any of them, a new outlier micro-cluster is created and it is placed in the outlier buffer.

This algorithm use fading window model for clustering the stream data. Core-micro-cluster is defined as CMC (W, C, R), W is the weight, C is center and R is radius. Algorithm for DenStream is as follows: DenStream (DS) : first define the minimal time span for micro cluster than get the next point at current time from data streams than merging process is done on data streams. In merging process first we try to merge point into nearest micro cluster if it does not fit into micro cluster than we try to merge it with outliers and check the weight of current micro cluster [3]. This process gets repeated if request of cluster is arrived and generate the cluster. DenStream algorithm does not release any memory space by either deleting a micro-cluster or merging two old microclusters [3].

B. HDDStream Algorithm

This algorithm is for clustering high dimensional data streams. In the online phase of this algorithm keep the summary of the points and dimensions and offline phase generates the final cluster based on given projected cluster [1]. Main three parts of HDDStream algorithm: first initial set of microcluster is extracted. After that online microcluster maintenance as new point has arrived and old points are expire due to ageing. Adding method is used in this phase where update dimension preference and find the closest micro cluster are there and final cluster is extracted on demand [1]. HDDStream can cluster high dimensional data effectively but in the pruning time it checks only micro cluster weights whereas micro cluster weight should be checked also. This is the disadvantage of the HDDStream algorithm.

The online phase keeps summarization of both points and dimensions and the offline phase generates the final clusters based on a projected clustering algorithm called PreDeCon. The algorithm uses DenStream concepts; however, it introduces prefer vector for each micro-cluster which is related to prefer dimension in high-dimensional data. A prefer dimension is defined based on variance along this dimension in micro-cluster. A micro-cluster prefers a dimension if data points of micro-clusters is more dense along this dimension. The micro-cluster with preferred vector is called a projected micro-cluster. Projected term shows that the micro-cluster is based on a sub- space of feature space and not the whole feature space. Based on this concept, the algorithm changes the potential and outlier micro-clusters to projected potential micro-clusters and projected outlier micro-clusters respectively. HDDStream has pruning time similar to DenStream in which the weights of the micro-clusters are periodically checked.

Merits and Limitations. HDDStream can cluster high-dimensional data stream; however, in the pruning time it only checks micro-cluster weights. Since the micro-cluster fades over time the prefer vector should be checked as well because it may change over time.

C. LeaDenStream Algorithm

In LeaDenStream, when a new data record x arrives, it is added to the Mini-Micro or Micro leader cluster based on the distribution of data in AdjustingLeader-Clusters. Then, we periodically and in every gap time, which is the minimum time for converting a dense mini-micro leader to a sparse, convert sparse minimicroleader clusters to dense and vice versa. We remove the sparse mini micro and micro leader clusters in PuringLeaderClusters.

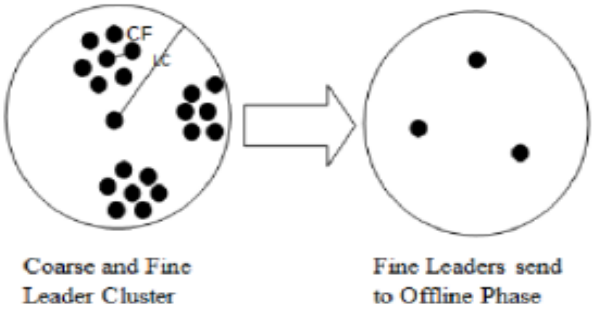

Fig.2: Mini micro and Micro Leader clusters

Online Phase: This phase is triggered when a data point arrives from data streams. The procedure is described as follows:

1. We try to find the nearest micro leader cluster to the data point
2. If we find such a micro leader cluster, we try to find nearest mini-micro leader cluster to the data point.
   a. If there is such a mini-micro cluster leader then merge the data point to the nearest mini-micro cluster leader.
   b. Otherwise, form a new mini-micro cluster with x as the center of new mini-micro cluster.
3. Otherwise, there is not such micro leader cluster, form a new micro leader cluster with x as the center of new micro leader cluster.

Furthermore, we prune the mini-micro and micro leader clusters in the gap time, Puring Leader Clusters. In the pruning time, all the micro leader clusters and their Mini Micro Cluster Leaders are checked. Micro and mini-micro leader clusters are kept in the tree structure to make it easier for searching and updating. Based on different kinds of Mini Micro Cluster inside micro cluster different decisions are made for pruning, which are described as follows:

- All the mini-micro leader clusters are dense: micro leader cluster center is kept for the offline phase
- All the mini-micro leader clusters are sparse: mini micro leader clusters are removed as well as their micro leader cluster.

- Some of mini-micro leader clusters are dense and some of them are sparse:
  1. Remove the sparse mini-micro leader clusters
  2. Keep the center of the dense mini-micro leader clusters for the offline phase

III. COMPARISON

This section compares the density-based micro-clustering algorithms discussed in the literature review based on several parameters.Table II shows the data that can be handles by the Density-based Micro-clustering Algorithms

Table III Data Handled by Density Micro-clustering Algorithms

Algorithm	Noisy data	Evolving data	Higher dimensional data
DenStream	Applicable	Applicable	Not applicable
HDDStream	Applicable	Applicable	Applicable
LeaDenStream	Applicable	Applicable	Not applicable

Table III shows the comparison of the resulting observations obtained from the review on the Density-based Micro clustering Algorithms. Table IV shows the time and space complexity of the density based micro-clustering algorithms

Table IIIII Results of Density Micro-clustering Algorithms

Algorithm	Limited Time	Limited Memory	Result
DenStream	Yes	yes	Arbitrary shape cluster
HDDStream	No	Yes	Arbitrary shape cluster
LeaDenStream	Yes	Yes	Arbitrary shape cluster

Table IVV Time and Space Complexity of Density Micro-clustering Algorithms

Algorithm	Space complexity	Time complexity
DenStream	M	O(M)
HDDStream	M	O(M)+O(Mp)
LeaDenStream	M	O(MMLC).O(MLC)

The literature on density-based clustering for data streams is usually centered around concrete methods rather than application contexts. Nevertheless, in this subsection, we would like to bring examples of several possible scenarios where density-based clustering can be used. Table V summarizes the review on Density-based Micro-clustering Algorithms

Table V Summary on Density Micro-clustering Algorithms

Algorithm	Type of data	Quality metric	Application domain
DenStream	Continous	purity	Network intrusion detection system
HDDStream	Continous	Purity	Environment monitoring
LeaDenStream	continous	Purity	Network intrusion detection system

IV. CONCLUSIONS

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 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

REFERENCES

[1] Ntoutsi I, Zimek A, Palpanas T, ”Density-based projected clustering over high dimensional data streams,” in Proc. the 12th SIAM Int. Conf. Data Mining, April 2012, pp.987-998.

[2] Amineh Amini, Teh Ying Wah, ”Leaden-Stream: A Leader Density-Based Clustering Algorithm over Evolving Data Stream,” in Journal of Computer and Communications, October 2013,volume 1, pp.26-31.

[3] Cao F, Ester M, QianW, Zhou A, ”Density-based clustering over an evolving data stream with noise,” in Proc. the 2006 SIAM Conference on Data Mining, April 2006, pp.328-339.

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

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

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

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

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

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

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

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

