International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) ISSN: 2395-5325
Volume 3, Special Issue 3 (September '2017)
Proceeding of - International Conference on Recent Trends in Computational Life Science and Information Technology,
Conference held at Tiruppur Kumaran College for Women,Tirupur, Tamilnadu,India.

AN EFFICIENT DECISION TREE BASED MICRO-CLUSTERING APPROACH USING DBSTREAM

M.Kavitha,
Assistant Professor
PG and Research Department of Computer Science,
Tiruppur Kumaran College for Women,
Tirupur,Tamilnadu,India.

R.Baby,
Research Scholar,
PG and Research Department of Computer Science,
Tiruppur Kumaran College for Women,
Tirupur,Tamilnadu,India.

Abstract: Clustering is a relevant task in mining data streams, which merge similar items in a cluster. Many clustering approaches are introduced in recent years for data streams that are based on density which provides protection against anomalies. Micro-clustering is a technique in stream clustering that stores the compact information of the data objects in data streams. Micro-cluster is a temporal extension of the cluster feature, which compresses the data effectively. The proposed system works a without limitation based algorithm with the purpose of automatically adapts to the speed of the information stream. It makes greatest use of the time available under the present constraints to provide a clustering of the objects seen awake to that point. The proposed approach incorporates the age of the objects to reveal the greater importance of more modern data. In efficient and effective handling, here we introduce the Decision Tree based DBSTREAM micro cluster (DTDBS), a compact as well as self-adaptive index formation for maintaining stream summaries. We present solutions to handle very fast streams at some stage in aggregation mechanisms and propose novel descent strategies so as to improve the clustering effect on slower streams as long as time permits. Our experiments show that our approach is capable of conduct a multitude of different stream characteristics designed for accurate and scalable anytime stream clustering.

Keywords : Micro Cluster, Best First Traversal, DataStream Clustering, Density-based Clustering

I.INTRODUCTION

Real-time analysis and mining of data streams have attracted substantial amount of researches because data stream mining means extracting useful pattern from data. They need to be processed as they arrive. Clustering data streams have becomes more attention in data mining research. In clustering data streams, the goal is to group the data streams periodically so there is an up-to-date clustering of all objects. Clustering data streams has many challenges such as Single pass clustering in which data arrive contentiously so the clustering has to be completed in a single pass over the data. Another challenge is Limited time in which the algorithm has to handle the speed of data streams for clustering. Other challenges may be number of clusters, noisy data. Distance-based data stream clustering methods are able to find ellipsoid-shaped clusters, and at best convex clusters. These methods have trouble finding the true clusters for non convex clusters. Density-based clustering algorithms are designed to find clusters of arbitrary shape and to handle anomalies. In these clustering algorithms the high density area is differentiated from the low one. Micro-clusters method is used to record summary information related to the data objects in the streams. Several algorithms are such as developed that are used this micro-clusters approach for their clustering.

The density-based data stream clustering methods available adopts a two-phase scheme approach consisting of an online phase, in which raw data is worked out to collect summary statistics, The online component is utilized by the analyst who can use a wide variety of inputs (number of clusters, time horizon) in order to give a quick view of the broad clusters in the data stream and an offline phase that creates the clusters by applying the summary data.

II.RELATED WORK

Density-based clustering [1] is a well-researched area and we can only give a very concise overview here. DBSCAN [2] and numerous of its improvements can be seen as the prototypical density-based[3,4] clustering approach. DBSCAN estimates the density around every data point by counting the While many algorithms [16,17] have been introduced that begin the problem of clustering on evolving data streams, hardly any attention has been paid to appropriate evaluation measures. Measures developed for static scenarios, namely structural measures and ground-truth-based measures, cannot correctly reflect errors attributable to emerging, splitting, or moving clusters. These situations are natural to the streaming context due to the dynamic changes in the data supply. Into this paper we develop a novel evaluation measure for stream clustering called Cluster Mapping Measure (CMM) [20]. A lot of density-based clustering approaches were produced at the last of the previous decade, including DBSCAN [11], DENCLUE [12], OPTICS [13], and CLIQUE [14].DBSCAN[1] is for the clustering of large noisy datasets on spatial data. DBSCAN introduced the idea of neighborhoods as a region of given radius and has a minimum number of data points.. It develops clusters based on a density-based connectivity analysis. DENCLUE[2] is also focus on neighborhoods, based on high-dimensional multimedia databases. In the multidi- mensional data space, DENCLUE calculates the impact of a data point upon its neighborhood and applies it to assign data points to the clusters. It clusters the items based on a set of density

distribution functions.

III.PROPOSED APPROACH

Synthetic Data Generation

In the field of mathematical modeling, a radial basis function network is an artificial neural network that uses radial basis functions as activation functions. The outcome of the network is a linear merging of radial basis functions of the inputs and neuron parameters. Radial basis task networks have many uses, including function guess, time series prediction, classification, and system control. Radial basis function (RBF) networks normally have three layers: an input layer, a hidden layer with a non-linear RBF activation function and a linear output layer. The input can be created as a vector of real numbers $\mathbf{x} \in \mathbb{R}^n$. The output of the network is then a scalar.

$$\varphi(\mathbf{x}) = \sum_{i=1}^{N} a_i \rho(||\mathbf{x} - \mathbf{c}_i||)$$

where is the number of neurons in the hidden layer, is the center vector for neuron, and a_i is the weight of neuron i in the linear output neuron? Functions that depend only on the distance from a center vector are radially symmetric about that vector, hence the name radial basis role. In the basic form all inputs are joined to each hidden neuron. The form is typically taken to be the Euclidean distance (although the Mahalanobis distance appears to perform better in general) and the radial basis function is commonly taken to be Gaussian

$$\rho\big(\|\mathbf{x} - \mathbf{c}_i\|\big) = \exp\left[-\beta \|\mathbf{x} - \mathbf{c}_i\|^2\right].$$

The Gaussian basis functions are local to the center vector in that

$$\lim_{||x|| \to \infty} \rho(||\mathbf{x} - \mathbf{c}_i||) = 0$$

i.e. changing parameters of one neuron has only a small effect for input values that are far away from the center of that neuron.

Micro cluster feature

Micro-clusters are a popular technique in torrent clustering or scaling clustering to large data sets to create and maintain compact representations of the current clustering. Instead of storing all incoming objects, a cluster feature tuple CF = (n, LS, SS) of the number n of represented objects, their linear sum LS, and their squared sum SS is maintained. This tuple suffices for computing mean and variance and can be incrementally updated. Any cluster feature (CF) then represents a micro-cluster, i.e., a set of objects and the main characteristics of its distribution. With this, objects can be easily assigned to the most similar micro-cluster incrementally. Existing micro-cluster approaches lack support for varying stream inter-arrival times. It is therefore crucial to provide the means for anytime clustering and self-adaptation to stream speed. We propose maintaining huddle features (CFs) by extending index structures from the R-tree family. Such hierarchical indexing structures provide the means for efficiently locating the right place to insert any object from the stream into a micro-cluster. The idea is to build a ladder of micro-clusters at different levels of granularity. Given enough time, the algorithm descends the hierarchy in the index to reach the leaf entry that contains the micro-cluster that is most similar to the current object. If this micro-cluster is similar enough, it is updated incrementally by this object's values. Otherwise, a new micro-cluster may be formed.

Micro Cluster tree

The tree is created and updated like any multidimensional index such as R-tree, R*-tree, etc. Distinct the minimum bounding rectangles that they maintain in addition to the objects, we store only CFs in the micro cluster tree. For insertion, we descent into the subtree with the closest mean with respect to Euclidean distance. Splitting is based on pairwise distances between the entries, where entries are combined into two groups such that the sum of the intra-group distances is minimal. The important property that reflects anytime capability of the micro cluster tree is its buffer in each entry. It serves as a temporary storage place of aggregates or matter that do not reach leaf level during insertion. Whenever insertion is interrupted, the recent CF is simply stored in the buffer of the entry that corresponds to the sub tree into which to descend next. This makes sure that future descent down the same subtree is used for continuing the insertion process. Whenever the descent destinations of the current insertionCF and the hitchhiker differ, the latter is placed in the corresponding buffer again to wait for the next ride down the tree.

Date Clustering Maintaining

In order to maintain an up-to-date view, we would like new objects to be more important than older objects. For this basis, the half-life of objects is 1 ë. To include decay, temporal information has to be added to the micro cluster tree nodes. The buffer is also added to exactly one of the child entries' cluster features.

Stream Aggregation

we propose a speed-up during aggregation before insertion: If we do not insert each object individually, there is more time to descend deeper with an total of objects. Naively, one could add up a certain number m of incoming objects, insert the total; sum up the next m objects, and so on. Clearly we need to keep fit some control over which objects should literally "go together". Ideally, we want to aggregate objects that would end up in the same leaf if we could descend the tree with them. Most probably, the new objects are not all similar to each other, but we expect subgroups with inter-similarity council of the clusters we also find in the tree.

Priority breadth first traversal

The alternative descent strategy is a priority breadth first descent. The single path depth first descent does not perform any kind of backtracking and thus cannot correct any misguided choice that might occur due to overlapping entries on higher levels of the tree. In other words, even if we chose the nearby entry on level k, we cannot be sure that

the nearby entry on level k + 1 is one of its children. To assure result the closest entry on each level, we therefore propose to evaluate all entries per point. The priority breadth first descent instead of checking a single node on each level as in depth first descent, each level is evaluated to find the closest entry by going through a list of entries sorted by the distance between the parent node and the present insertion object.

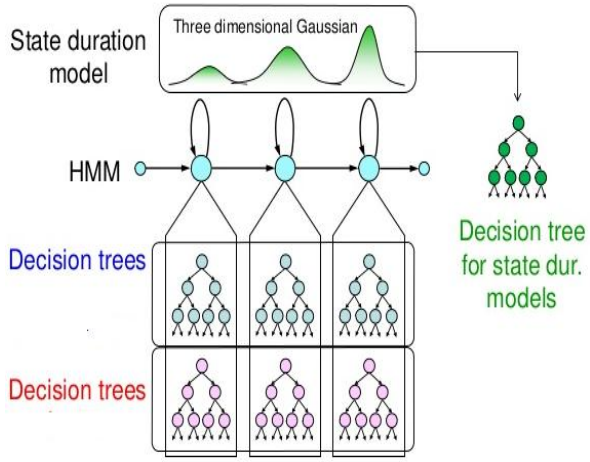

Fig : 2 Micro Clustering Using Decision Tree

Best first traversal

The priority queue contains the entries seen so far that have not been refined yet and their corresponding distance to the insertion object. Given the time to make the next step in descending down the tree, the best first approach always takes the first element from the queue, i.e., the entry which has the smallest reserve to the object. The distances from the object to the entries in the corresponding kid node are computed and inserted into the priority queue (refinement). This process continues until addition is interrupted or until all nodes have been visited. As in the priority breadth first descent we keep the property of anytime clustering and update the path with cluster features (CFs) when we buffer or insert the object on interruption.

Iterative depth first descent

In terms of anytime clustering, best first descent tries to optimize the selection of insertion nodes. A possible drawback of this strategy is that depending on how often the algorithm has to go back and continue from upper nodes and on how soon it is interrupted, the algorithm might remain at the upper levels and buffer the object there. Similar drawbacks can be expected from the priority wideness first strategy. In contrast, depth first processing is most often able to reach leaf level.

Cluster shapes and cluster transitions

This can be seen as our online component and it allows for using various offline clustering approaches. Taking the means of the CFs as council, we can apply a k-center clustering or density based clustering as proposed to detect clusters of arbitrary shape. One main advantage of our approach is that we can maintain a way larger number of micro-clusters compared to other approaches and hence the offline clustering, e.g. density based, has finer input granularity.

IV. DBStream:

DBStream is a density-based clustering approach for data streams. It uses the micro-cluster concept. Micro-cluster density is made on weighting areas of points in the neighborhood. It uses the fading function and aggregates the weights on micro-clusters. DBStream consists two phases -online-offline- and is fully based on CluStream framework. Online Phase**:** The p-micro- clusters and o-micro-cluster are placed in an online manner. All the o-micro-clusters are kept in a separate memory space, which is called outlier-buffer. When a new point arrives, it will be joined with the existing micro clusters.

Offline phase: After finding the p-micro clusters in the online phase, a variant of DBSTREAM algorithm is used to p-micro-clusters to get the final result of clustering. Each p-micro-cluster is seen as a virtual point for clustering. Weight is a needed parameter in this variant DBSTREAM. For any decision, the weight factor is verified, which must be higher than the threshold.

V. EXPERIMENTAL RESULT

In this section, the result of proposed algorithm with various attributes selection with UCI Repository datasets and information gain on some datasets with some continuous attributes are presented. Accuracy and execution time comparison for block incremental clustering is done. Accuracy and execution time comparison for different attribute selection measure is presented Experimental setup to perform the proposed method, java is used in Netbeans8.2 is used as front end to provide efficient storage structure. For the preprocessing task like missing value replacement weka tool is used, before applying to these algorithms. We used UCI repository WPBC Datasets.

The real-life data set, named Wisconsin Breast Cancer dataset is used. The data set is publicly available on UCI machine learning repository and consists of 699 instances with nine continuous attributes.

Dataset	Precision	Recall	Overall Accuracy (%)*100
WPBC	0.694	0.312	0.62
Liver Dataset	0.612	0.214	0.68
Lung Cancer dataset	0.712	0.211	0.73

The processing time of *DBStream* increases linearly as the stream proceeds. The execution time is evaluated on data stream with different dimensionally and different number of natural clusters. The empirical result figure outs that the algorithm is more efficient than others.

VI. CONCLUSION

Clustering data streams produces additional constraints on clustering algorithms. As data is constantly arriving, time for meting out is limited. Clustering has to be performed in a single pass over the incoming data and within the possibly anecdotal inter-arrival times of the stream. also, memory is limited, making it impossible to store all data. For clustering, we are faced with the challenge of maintaining a current result that can be presented to the user at any given time. The proposed system works a without parameter based algorithm that automatically adapts to the speed of the data stream. It makes best use of the time available under the current constraints to provide a clustering of the objects seen up to that point. The proposed approach incorporates the age of the objects to reflect the greater importance of more recent data.

For efficient and effective handling, here introduce the Tree based DBSTREAM micro cluster (TDBS), a compact and self-adaptive index structure for maintaining stream summaries. moreover, we present solutions to handle very fast streams through aggregation mechanisms and propose novel descent strategies that improve the clustering result lying on slower streams as long as time permits. This experiment result out that our approach is capable of handling a multitude of various stream self for accurate and scalable anytime stream clustering.

VII.REFERENCES

[1]. S.Guha, N.Mishra, R.Motwani, and L. O'Callaghan, "Clustering data streams," in Proc. ACM Symp. Found. Comput. Sci.,
[2]. K. Udommanetanakit, T. Rakthanmanon, and K. Waiyamai, "Estream: Evolution-based technique for stream clustering," in Proc. 3rd Int. Conf.Adv. Data Mining Appl., 2007, pp. 605–615.
[3]. F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in Proc. SIAM Int. Conf.Data Mining, 2006, pp. 328–339
[4]. Y. Chen and L. Tu, "Density-based clustering for real-time stream data,".
[5]. D. Tasoulis, N. Adams, and D.Hand, "Unsupervised clustering in streaming data,".
[6]. K. Udommanetanakit, T. Rakthanmanon, and K. Waiyamai, "Estream: Evolution-based technique for stream clustering," in Proc. 3rd Int. Conf.
[7]. Adv. Data Mining Appl., 2007, pp. 605–615.
[8]. A.Amini and T. Y. Wah, "Leaden-stream: A leader density-based clustering algorithm over evolving data stream," J. Comput. Commun., vol. 1, no. 5, pp. 26–31, 2013.
[9]. M.Hahsler and M.H. Dunham, "Temporal structure learning for clustering massive data streams in real-time," in Proc. SIAM Conf. Data Mining, Apr. 2011, pp. 664–675.
[10]. D.Tasoulis, N.Adams, and D.Hand, "Unsupervised clustering in streaming data," .
[11]. J.A. Silva, E.R.Faria, R. C.Barros, E. R. Hruschka, A. C. P. L. F. d.Carvalho, and J. A. Gama, "Data stream clustering: A survey," .
[12]. .M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise,".
[13]. A.Hinneburg and D. A. Keim, "An efficient approach to clustering in large multimedia databases with noise," in KDD, 1998, pp. 58–65.
[14]. M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: ordering points to identify the clustering structure," SIGMOD Rec., vol. 28, pp. 49–60, June 1999.
[15]. R.Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," SIGMOD Rec., vol. 27, pp. 94–105, June 1998.
[16]. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: Massive online analysis," J. Mach. Learn. Res..
[17]. H.Kremer, P.Kranen, T.Jansen, T. Seidl, A. Bifet, G. Holmes, and B.Pfahringer, "An effective evaluation measure for clustering on evolving data streams," .
[18]. J. Gama, R. Sebasti~ao, and P. P. Rodrigues, "On evaluating stream learning algorithms," Mach. Learn., vol. 90, pp. 317–346, 2013
[19]. Bifet, G. de Francisci Morales, J. Read, G. Holmes, and B.Pfahringer, "Efficient online evaluation of big data stream classifiers," in Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2015, pp. 59–68.
[20]. C.C.Aggarwal, et. al., "A framework for projected clustering of high dimensional data streams," in Proc. Int. Conf. Very Large Data Bases, 2004, pp. 852–863.
[21]. ZhongJi, Memberetr., et., al., "Relevance Preserving Projection and Ranking for Web Image Search Reranking" IEEE transactions on image processing, vol. 24, no. 11, november 2015