International Journal of Computer Applications (0975 – 8887)
Volume 116 – No. 17, April 2015

Multi-Density based Incremental Clustering

Lanka Pradeep
Department of computer science and systems engineering,
Andhra university college of Engineering

A.M.Sowjanya, Ph.D
Department of computer science and systems engineering,
Andhra university college of engineering

ABSTRACT
Clustering is the task of grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups . It is a common technique for statistical data analysis, used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics. A major difficulty in design of modern clustering algorithms is that, new datasets are dynamically added to the existing large database and it is not efficient to perform data clustering on the entire database every time a new dataset is added to the database. The new data added dynamically to the existing database is called incremental data. DBSCAN is widely used density based clustering algorithm. However it is known that DBSCAN fails to identify clusters of different densities. This paper presents a simple and efficient algorithm that identifies clusters of different densities and arbitrary shapes with automatic Eps estimation. Eps is estimated by using distance curve and difference of slopes and DBSCAN is applied on the data for each estimated Eps, resulting in multi-density clusters. Then by making use of formed clusters, incrementally updated data is clustered.

General Terms
Data mining.

Keywords
Data clustering, Incremental clustering, Multi-density clustering, Data mining.

1. INTRODUCTION
Data mining is the process of extraction of information from data set and transform it into an understandable structure for further use. Clustering is one of the models of data mining, which finds groups that are different from each other, and whose members are very similar to each other. It is a process in which a group of unlabeled patterns are partitioned into a number of sets so that similar patterns are assigned to the same cluster, and dissimilar patterns are assigned to different clusters.

The main problem with the conventional clustering algorithms is that, they mine static databases and generate a set of patterns in the form of clusters [12]. Numerous applications maintain their data in large databases or data warehouses and many real life databases keep growing incrementally. New data may be added periodically either on a daily or weekly basis. For such dynamic databases, the patterns extracted from the original database become obsolete. Conventional clustering algorithms handle this problem by repeating the process of clustering on the entire database whenever a significant set of data items are added. The process of rerunning the clustering algorithm on entire database is time consuming and inefficient.

A solution to handle this problem is to integrate a clustering algorithm that functions incrementally [3].Incremental clustering algorithms permit a single or a few passes over the whole dataset to put the updated item into the cluster. With respect to the size of the set of objects, algorithms and number of attributes, incremental clustering algorithms are of scalable nature [4].

This paper presents a simple and efficient clustering algorithm that is based on the density based DBSCAN [1] clustering algorithm. The algorithm identifies arbitrary shaped and multi-density clusters by estimating Eps parameters of DBSCAN automatically and iterating DBSCAN for each Eps. Eps is estimated by using distance curve [2] and difference between slopes of distance curve. Once the initial clusters are formed incremental data is clustered by making use of formed initial clusters. For each data point in the incremental data the distance between the point and core points of initial clusters is computed. The nearest core point is picked. If the distance between the incremental point and the core point is less than the density radius (Eps) of core point then the incremental point is added to the cluster belonging to core point otherwise it is considered as noise point.

2. RELATED WORK
Clustering algorithms may be classified into partitioning, hierarchical, density, model based and grid based methods [8]. Partitioning algorithms are k-means and k-medoid [8][9]. Hierarchical algorithms create a hierarchical decomposition of a database, e.g. single-link, complete-link, average-link method, BIRCH and CURE [8], [9]. Model-based clustering algorithms attempt to optimize the fit between the given data and some mathematical models, e.g. decision trees and neural networks. Density-Based Clustering algorithms group objects according to specific density objective functions, e.g. DBSCAN [1]. The proposed algorithm is based on the idea of DBSCAN.

The DBSCAN [1] is a base algorithm of density based clustering. It requires user to specify two global input parameters i.e. MinPts and Eps. The density of an object is the number of objects in its Eps-neighborhood of that object. DBSCAN does not specify upper limit of a core object. So due to this, the clusters detected by it, are having wide variation in local density and forms clusters of any arbitrary shape. DBSCAN starts with an arbitrary point p and retrieves all points' density-reachable points from p wrt. Eps and MinPts. If p is a core point, this procedure yields a cluster wrt. Eps and MinPts. If p is a border point, no points are density-reachable from p and DBSCAN visits the next point of the database. It takes the set D, Eps, minpts as input and labels each point with a cluster id or rejects it as a noise [1], [8]. Due to a single global parameter Eps, it is impossible to detect some clusters using one global-MinPts and Eps value. It does not perform well on multi-density data sets.

KDDClus [2] algorithm utilizes the KD-tree data structure for efficient processing in high dimensions. However it is expensive. It computes the kth nearest neighbor distance for each point during the distance computation using KD-tree data structure. The patterns corresponding to noise are expected to have larger k-distance values. The aim is to determine the knees for estimating the set of Eps parameters. This Eps value will be accepted from the user through interaction. A knee corresponds to a threshold where a sharp change of gradient occurs along the k-distance curve. This represents a change in density distribution amongst patterns. Any value less than this density-threshold Eps estimate can efficiently cluster patterns whose k-NN distances is lower than that, implying patterns belonging to a certain density. Analogously all knees in the graph can collectively estimate a set of Eps's for identifying all the clusters having different density distributions.

3. PROPOSED ALGORITHM

3.1 Identifying initial clusters

Initial clusters are identified by using DBSCAN on the database for each estimated Eps's.

3.1.1 Estimating range of Eps's

To determine different range of Eps values automatically and to identify the number of clusters of different densities, first a k-dist graph for all the points is drawn, for a given k which will be entered by the user. Initially the average of the distances of every point to all k of its nearest neighbors is computed [2]. These averaged k-distances are plotted in an ascending order and the knees [2] for estimating the set of Eps values are determined.

A knee corresponds to a threshold where a sharp change of gradient occurs along the k-distance curve [2]. This represents a change in density distribution amongst points. Any value less than this density threshold Eps estimate can efficiently cluster patterns whose average k distances is lower than that. To find all possible Eps values, slopes at regular interval are calculated and then the difference between slopes values at the same regular interval are found. By setting certain threshold value different Eps values can be found automatically based on this threshold value while discarding those with higher values than threshold.

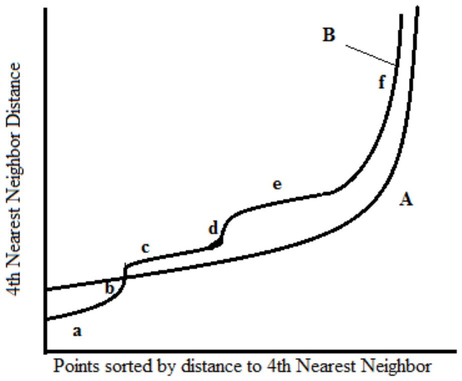

Fig 1: Sample k-dist plot

Figure 1 shows a sample k-dist plot. Line A shows a sample k-dist line of a single-density dataset while line B shows a sample line of a three varied-densities dataset. The shape of the sorted k-distance plot depends on the distribution of the k-nearest neighbor distances. The plot will look more "stairs-like" if the objects are distributed regularly within clusters of very different densities [2]. For points that are not in a cluster, such as noise points, the corresponding k-dist line rockets, connecting two smooth curves which stand for two density levels. Line b and d in Figure 1 are such lines. Line a, c and e represent three density regions of the dataset. For different density regions suitable eps is selected. For example, in Figure 1, there are three density levels. Line a shows the densest density level and e shows the sparsest one. Combine line a and b as a sub-k-dist plot to select Eps1, and then take line c and d as a sub-k-dist plot for Eps2, e and f for Eps3 finally.

Table 1. Algorithm for estimating Eps's

```
/* d: the sample dataset
   k: number of nearest neighbors to be considered for eps estimation */
EpsDBSCAN(d, k)
for each point p ∈ d
    N_p[ ]= GetNeighbors (p, k)
    for each point p' ∈ N_p
        Distance=Σ GetDistance(p,p')
    Dist[]=Distance/k

SortAscending(Dist[])
Plot Dist[] on y axis

for(int i=0;i<d.size();i+=k)
    SlopeValues[]= GetSlope(Dist[i],Dist[i+k])
    DistValueAtSlope[]=Dist[i+k]

for(int i=0;i<SlopeValues.size();i++)
    SlopeDifference=SlopeValues[i]-SlopeValues[i+1]
    If(SlopeDifference > threshold * SlopeValues[i])
        Eps[ ]=DistValueAtSlope[i]

return Eps[ ]
```

3.1.2 Multi-density clustering

Clusters having different densities or intensity are called multi density clustering [2]. After determining the number of different Eps values clusters can be formed starting from the lowest Eps value in the sorted k-dist graph, by iteratively executing DBSCAN for each of the Eps estimated considered in ascending order marking the points in the already detected clusters as "visited". The value of minpts is taken equal to the value of k in k-dist plot which is a user input parameter. In this manner all clusters are determined in a multi-density framework, in a decreasing order of density, with noise being modeled as the sparsest region [2].

Table 2. Algorithm for multi-density based clustering

```
/* d: the sample data set.
   Eps's: the estimated Eps's in ascending order
   Minpts: the value of Minpts is equal to k in k-dist graph*/
MULTIDBSCAN(d, Eps's,Minpts)
  for each Eps ∈ Eps's
      objs =cluster from DBSCAN(d, Eps,Minpts)
      remove objs from d
  mark d as noise objects
```

3.2 Clustering incremental data

Once the initial clusters are identified the incremental data is clustered based on the initial clusters. A cluster contains at least one core point [1]. For each unlabeled incremental data point, find the closest core point. If their distance is less than the Eps of the core point then the unlabeled instance is considered to be in the same cluster as the core point. Otherwise, it is labeled as noise. As density of clusters may differ in multi-density clustering, core points belonging to different clusters may have different Eps values and core points belonging to same cluster have same Eps value.

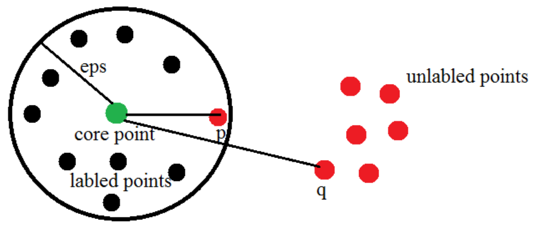

Fig 2: Assignment of incremental data

For example, in figure 2 point q is considered as noise point because the distance between q and core point is more than the Eps of core point. Similarly point p is labeled into same cluster as core point because the distance between p and core point is less than Eps of core point.

If the number of noise points are equal to MinPts, then a new cluster is formed comprising of these noise points. The point having the least kth nearest neighbor distance among these noise points is selected as core point of the newly formed cluster, where k is equal to value of MinPts. The Eps of the core point is equal to the kth nearest neighbor distance.

For example, in figure 3 Minpts is equal to 7. The point C is selected as core point of the newly formed cluster because it has the least kth nearest neighbor distance, in this case k is 7.Value of eps is least kth nearest neighbor distance from point C, where k is 7 in this case.

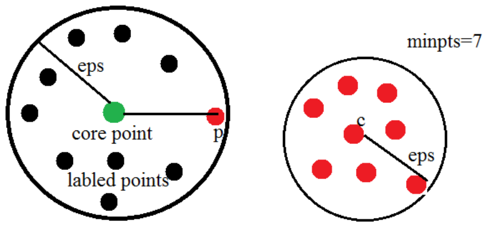

Fig 3: Forming new cluster

This process is repeated for every point in the incremental data.

Table 3. Algorithm to cluster incremental data

```
/* $d_i$: incremental dataset
   C: set of core points
   Eps: set of eps's
   Minpts: equal to value of k entered by user in previous
steps.
*/
IncrmentalCluster($d_i$, C, Eps, Minpts)
for each point p ∈ $d_i$
    find some C[x] such that dist(C[x],p) is minimum
    if(dist(C[x],p) < = Eps[C[x]])
        add p to cluster belonging to C[x]
    else
        add p to Noise
    if(count(Noise)>=minpts)
        K=minpts
        form new cluster M
        C[n+1]=m such that dist(m,$m_K$) is minimum
        Eps[C[n+1]]=dist(m,$m_K$)
        Where m,$m_K$∈ M
```

4. EXPERIMENTAL RESULTS

Experiments were carried out on Intel core i5 2.40GHz processor with 4GB RAM, 64 bit windows 8 operating system. Programs were implemented in C++. Algorithm was applied on Compound dataset obtain from research department of The University of EDINBURGH Schools of informatics specially provided for testing new algorithms. To convert the static database into dynamic half of the points are used for initial clustering and the remaining points are considered as incremental data points.

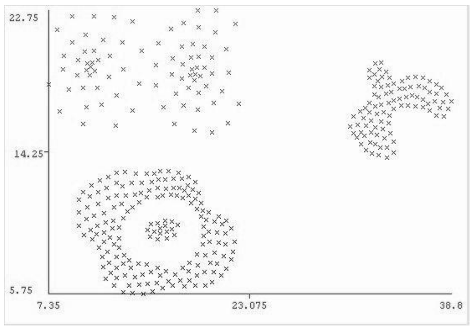

Fig 4: Dataset used for experiment

Half of the data set is used for initial cluster formation and the other half is given as input to algorithm as incremental data. Figure 5 shows initial cluster formation with K=5. Where K is number of nearest neighbors.

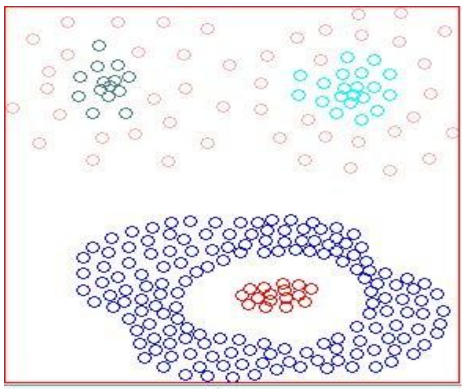

Fig 5: Initial clusters formed with k=5

After the initial clusters are formed, using the initial clusters incremental data is clustered. The result of incremental clustering is as shown in figure 6.

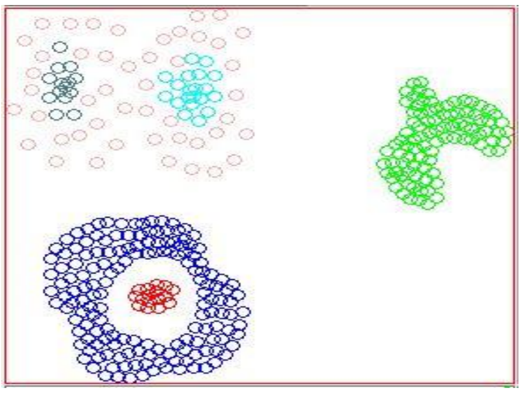

Fig 6: Incremental data clustering

The green cluster in figure 6 comprises of the incremental data. Although the algorithm clusters the incremental data efficiently it fails to distinguish noise present in the incremental data in some cases.

5. CONCLUSION

This paper presents a simple and efficient algorithm for clustering incremental data based on density. The proposed approach has two modules namely, initial clustering and incremental clustering. First the initial cluster are obtained by sequentially applying dbscan algorithm with various eps values estimated and then incremental data is clustered based on the initial clusters. Algorithm was applied on Compound dataset obtain from research department of The University of EDINBURGH Schools of informatics. The experimental results indicate that the proposed algorithm identifies clusters efficiently. Future scope will be to find an efficient way to identify noise in the incremental data, which the algorithm fails to identify in some cases and to find the value of K internally thus making the entire process automatic.

6. REFERENCES

[1] M Ester, H-P. Kriegel. J. Sander, and X, Xu. 1996. “A density-based algorithm for discovering clusters in large spatial databases”. KDD’96.

[2] Sushmita Mitra, Jay Nandy, KDDClus: A simple method for multi-density clustering, in: Proc.2011 International Workshop on Soft Computing Applications and Knowledge Discovery (SCAKD’2011), Moscow, 2011, 72-76.

[3] Chien-Yu Chen, Shien-Ching Hwang, and Yen-Jen Oyang,"An Incremental Hierarchical Data Clustering Algorithm Based on Gravity Theory", Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, Pages: 237 - 250, 2002.

[4] Dan Simovici, Namita Singla, "Metric Incremental Clustering of Nominal Data", ICDM, pp: 523-526, 2004.

[5] M. Charikar, C. Chekur, T. Feder, and R. Motwani, Incremental clustering and dynamic information retrieval,in Proc. of the 29th Annual ACM Symposium on Theory of Computing, 1997, pp. 626–635..

[6] Jiawei Han, Micheline Kamber, “Data Mining Concepts and Techniques”, Harcourt India Private Limited, 2001.

[7] Seokkyung Chung and Dennis McLeod, "Dynamic Pattern Mining: An Incremental Data Clustering Approach",Journal on Data Semantics, Vol. 2, pp. 85-112, 2005.

[8] Pang-Ning Tan, Michael Steinbach, Vipin Kumar, “Introducing to Data Mining”, Pearson Education Asia LTD, 2006.

[9] Jason D. Peterson, “Clustering overview”, http://www.cs.ndsu.nodak.edu/~jasonpet/CSCI779/Clustering.pdf.

[10] Wikipedia “cluster analysis”, http://en.wikipedia.org/wiki/Cluster_analysis.

[11] M. H. Marghny, Rasha M. Abd El-Aziz and Ahmed I. Taloba, “An Effective Evolutionary Clustering Algorithm: Hepatitis C Case Study”, Computer Science Department, Egypt, International Journal of Computer Applications, vol. 34, No.6, pp. 0975-8887, 2011.

[12] Sowjanya, A.M. and M. Shashi, 2010. Cluster featurebased incremental clustering approach (CFICA) for numerical data. IJCSNS Int. J. Comp. Sci. NetworkSecurity, 10.

[13] R. Ding, Q. Wang, Y. Dang, Q. Fu, H. Zhang, and D. Zhang. Yading: Fast clustering of large-scale time series data. In VLDB, 2015.